

# TACL 鬼

## The TACL GUI User Manual

[GUI version 1.0.0]

Michael Radich and Sharon Chi

## Contents

Acknowledgements.....	4
Technical requirements.....	4
Background.....	5
TACL-related methodology.....	18
The TACL GUI.....	18
GUI download.....	18
WORKFLOW STEP 1: Corpora.....	19
WORKFLOW STEP 2: Databases.....	22
Generating or importing a database in(to) the GUI.....	23
(1) Generate a database.....	24
(2a) Download the full database for the “Radich corpus” version of the Taishō.....	28
(2b) Import a database.....	28
(3) Select a database.....	30
(4) Update a database.....	31
(5) Delete a database.....	32
WORKFLOW STEP 3: “Catalogues”.....	33
Modifying the corpus, database and/or catalogue(s).....	35
WORKFLOW STEP 4: Running core TACL tests.....	37
Intersect.....	39
Difference.....	47
Asymmetric Difference.....	49
WORKFLOW STEP 5: Supplied Intersect.....	55
Search.....	59
WORKFLOW STEP 6: Filter/Rationalize:.....	63
WORKFLOW STEP 7: Working with the results (1): Import and sorting in Excel (or similar).....	72
WORKFLOW STEP 8: Working with the results (2): Examining results back in context.....	82
Beyond the GUI.....	84
Glossary.....	84
TACL-based research publications.....	87



Before you use the TACL GUI, we suggest that you first read through this manual in its entirety.

Use of the TACL GUI requires a certain amount of background knowledge, in terms of both how the tool works, and how best to conceive conceptually of its use for the study of problems in Chinese Buddhism. Users are asked to have some patience with the background learning required.

TACL-specific terms used in this Manual are listed in a Glossary at the end of the document.

### **Acknowledgements**

The TACL GUI was programmed on an entirely voluntary basis by [Roland Borsos](#) (PhD student, Heidelberg University). The source code for TACL was created by [Jamie Norrish](#). We very gratefully acknowledge these generous contributions of time and expertise.

Work on TACL and the TACL GUI has been supported by the Alexander von Humboldt Research Foundation, Victoria University of Wellington, Heidelberg University, and the Deutsche Forschungsgemeinschaft (DFG, project no. RA 3202/1-1). We gratefully acknowledge that support.

### **Technical requirements**

TACL is programmed in python (a programming language), but separate installation of python is not necessary—the GUI comes with python and all the necessary add-ons bundled in.

The TACL GUI is available for [Windows](#), [Linux](#), and [Mac](#) (the Mac version exists thanks to Merlin Beutlberger).

Windows users require Windows 10 or later (earlier versions of Windows are not compatible with the necessary version of python).

The disc space required depends on corpus size. For some research questions, it is possible to use the GUI with relatively small corpora. For a large corpus like the full Taishō, however, the TACL database can be quite large (e.g. 170 GB). Users working with a full database may need up to 200 GB of free disk space.

The generation of a large database can place heavy demands on the computing power of your system. The more RAM you have, the better; the faster your processors and read-write capacity to disk, the better. If your system is less powerful, you might expect longer computing times. The generation of a large database usually takes hours, at least, even on a powerful system, and could take several times longer on a less powerful computer. For this reason, some users may choose to

download the full database instead (see below). Either way, once you have a database, standard TACL processes (such as Intersects and Differences—see below) can also take minutes or hours, depending upon the size of the corpus under analysis.

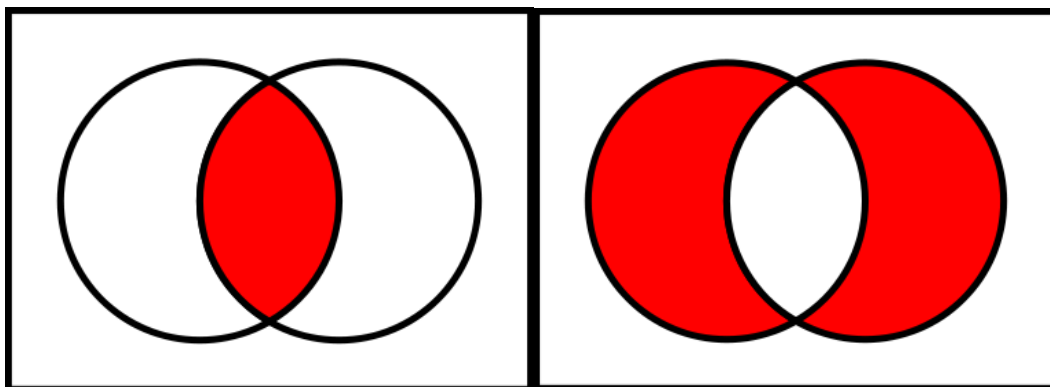
As we describe below, we have provided users with the option of downloading a full TACL database for the Taishō corpus. However, even in zipped form, the download file is large (ca. 47 GB). Depending upon the speed of your Internet connection, this initial download may also take some time and represent a technical hurdle.

As we will also describe below, we have bundled everything required for (the Windows version of) the TACL GUI, including the exercises in this Manual, into a "[TACL GUI starter kit](#)". Before we go into such detail, however, we will first lay out some basic background.

### **Background**

The TACL GUI, as the name suggests, is a GUI (Graphic User Interface, i.e. point-and-click interface of the type most users are used to for most computer programmes) for the use of "TACL". "TACL" is a tool for the computer-assisted analysis of digitised texts in the Chinese Buddhist canon. The name "TACL" is a "backronym" motivated mainly by a love of puns, and it doesn't really matter what it stands for. Interested users can contact us with suggestions for fun interpretations; users who prefer to have answers can think of it as meaning "Text And Corpus Lab".

Norrish and Radich began working on TACL at the end of 2012. It was originally conceived as a tool allowing the simple, large-scale comparison of Chinese Buddhist texts or corpora. More specifically, at the conceptual heart of the tool are two comparative operations: (1) Discover all material shared by the texts/corpora under analysis; or (2) discover all material distinctive to one text/corpus, against the other(s). In other words, TACL finds *intersects*, or *differences*.

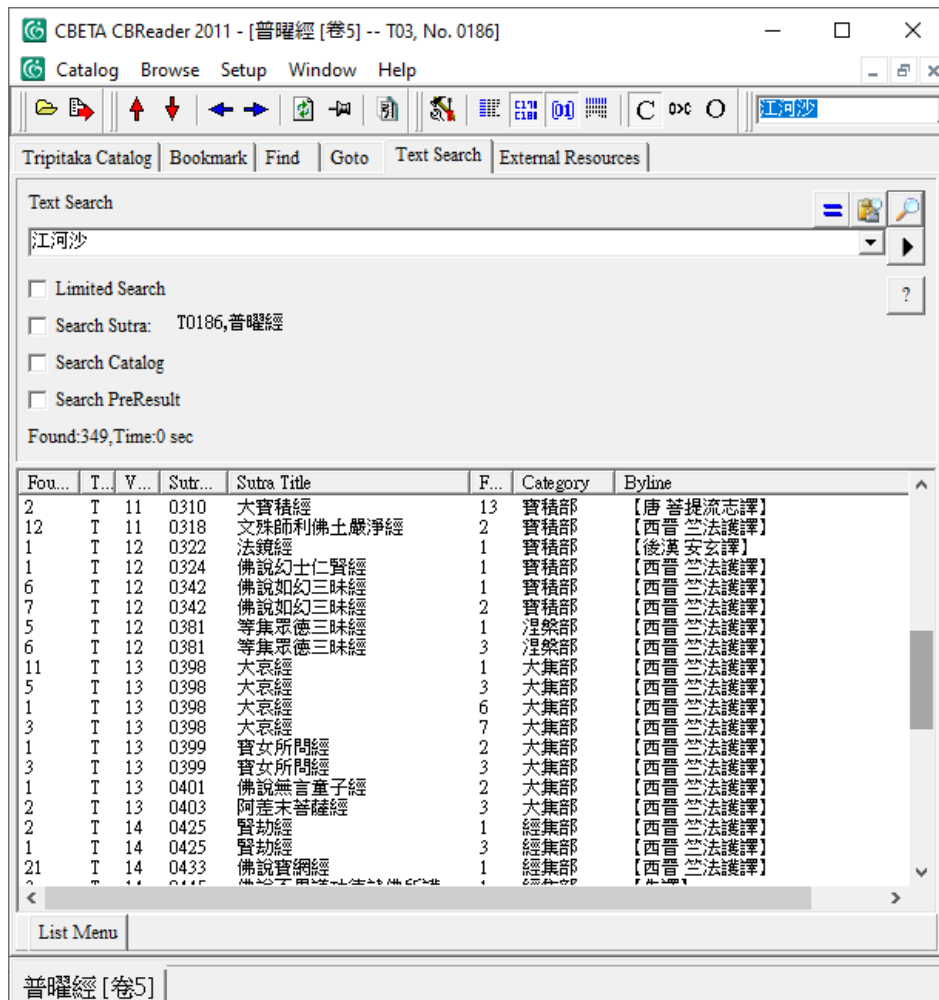


(It really is that simple.)

Here, "material" (shared or distinct) means more specifically: n-grams, i.e. strings (series of contiguous tokens = Chinese characters). Here, *n* is the length of the string. For example, 如來 is a 2-gram; 摩訶般若波羅蜜 is a 7-gram.

Users should remember this basic fact about the tool—that it works with n-grams—in order to understand the significance of what it finds. This fact implies a couple of caveats. TACL finds only *contiguous, literal strings*, matching the *pattern of distribution* stipulated by the type of test selected, and the *texts/corpora defined as input*. "Contiguous" means that TACL cannot find any pattern or turn of phrase, however obvious it is to a human reader, if other text or tokens intervenes between two mutually remote parts of the phrasing—e.g. it cannot find such patterns as 與...具 or 以...故. "Literal" means it does not know that two things are the same unless they are written (i.e. encoded) exactly the same—e.g. it does not know that 燕坐 is in some contexts identical in meaning with 晏坐, or 燕座.

TACL can thus be conceived of as a kind of souped-up search tool. However, in both of these core operations—Intersect, and Difference—TACL differs from a search tool like the CBReader in one key respect. In a CBReader-type search tool, the user inputs a *string*, and outputs the *pattern of distribution* for that string. For example, the user says that she wants to know where 江河沙 occurs, and gets something like this (showing that the string is heavily concentrated in Dharmarakṣa):



In TACL, by contrast, the user decides what *pattern of distribution* they are looking for, and the tool returns *all* strings that match that condition. "TACL-shaped questions" thus look like these examples: "What strings occur in both Dharmarakṣa and Kumārajīva"? or, "What strings occur only in Dharmarakṣa, but never in any other texts in the canon?"

Systematic discovery of strings matching such patterns helps us discover evidence for the treatment of various research questions hinging on intertextual relations. Most typically, *Intersect* tests can be applied to study the prior sources or subsequent impact of a given text; and *Difference* tests can be applied to discover stylistic traits distinctive of a certain corpus. The TACL GUI has been designed with these two applications primarily in mind. Examples of the application of TACL to real research questions may be seen in the publications listed in the Bibliography at the end of this Manual. Other simple examples will be introduced in the exercises described below.

Despite the deceptive conceptual simplicity of the core TACL operations of *Intersect* and *Difference*, the tool has considerable power. This power derives from two or three principal sources.

(1) First, individual tests can be "concatenated", or combined in series—in principle, *ad infinitum*. For example, we can ask: What strings are only ever found in Dharmarakṣa, and never in any other reliably ascribed translations? (a *Difference* test). But on that basis, we can then ask further: Which of these strings are *also* found in Text Q, of otherwise unknown or dubious ascription? (an *Intersect* test, combined with the results of the first Difference). Such concatenated tests can combine to give us much greater analytical power to address a research question than any single test in isolation. (Concatenation is achieved by the function we call WORKFLOW STEP 5: Supplied Intersect, which we introduce in more detail below.)

(2) Second, a computer can relatively rapidly chew through quantities of text that a human reader would struggle to analyse in a lifetime. Some TACL tests can still take a few hours, but in that time, they can work through the entire Taishō canon. Moreover, within its literalistic limits, a computer is perfectly accurate.

(3) Third, TACL ironically derives considerable power from the very blindness of the machine—especially when combined with the insights of an informed and creative human user. The user defines a pattern of distribution, as described above, and the machine discovers *all* strings that match that pattern. A human, by contrast, guessing what sorts of strings might sport the pattern of distribution in question, is far more likely to think only of things that already conform to existing knowledge, or conditioned patterns of attention. Experience with TACL has shown that precisely this blindness ironically helps us to *see* things we would otherwise overlook. Typical studies of problems of style and attribution in prior scholarship have tended to focus on phraseology with Buddhist conceptual content as evidence. With TACL, we are more easily prompted to notice also the evidential significance of Buddhologically "nondescript" items—a kind of phraseological "wallpaper"—which a human reader would be unlikely to notice unassisted. An eye-opening example is the perfectly complementary distribution of 村邑 and 村落 in the *Madhyamāgama* T26 and the *\*Ekottarikāgama* T125 respectively (see Radich and Anālayo 2017: 229).

In combination, these last two features of TACL make it like a powerful combination of a microscope, which discovers fine details invisible to the naked eye, and a telescope, which allows us to see further than our ordinary sight.

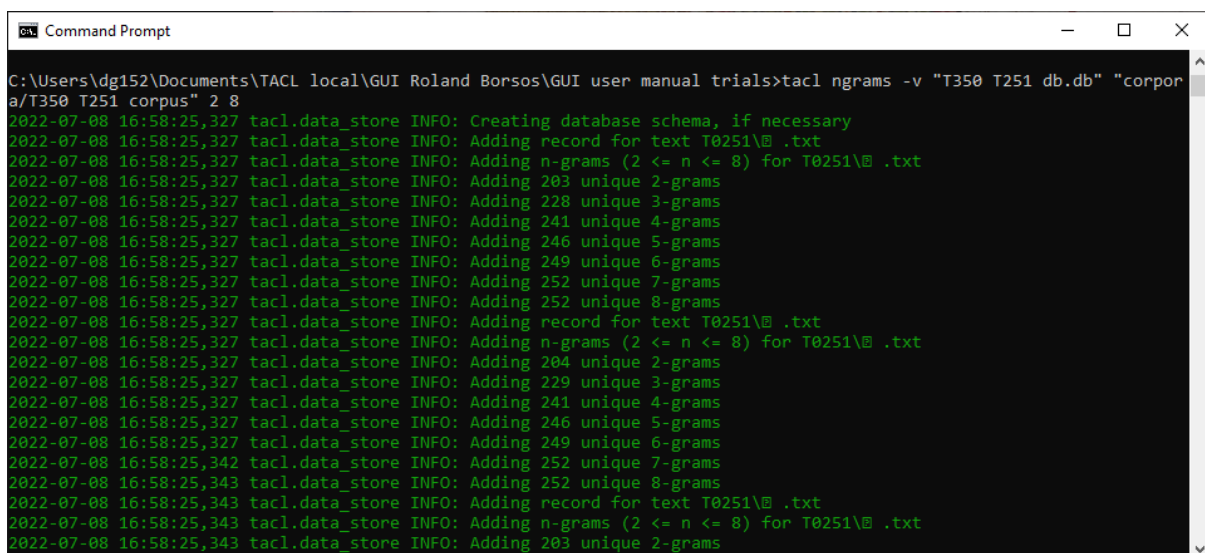
At the same time, it is important to remember that the power of TACL only comes from the *combination* of the machine and a trained, well-informed, carefully thinking human user. That is to say, TACL methods can usefully be thought of as "cyborg" methods—a partnership of human and machine. The machine alone can do no research, and it finds no answers. It finds only *potential* evidence for our research questions, which must then be weighed up by a philological mind. In this



sense, we might liken the tool to a metal detector, which might help an archaeologist confronted with a large site decide where to dig first. The tool suggests that interesting evidence might be in certain spots—but not everything we dig up might be useful (we might also get some junk). Equally, not everything useful will be found by the tool (there might be evidence made of materials other than metal, which we need other tools to find). A human researcher will have to analyse each piece of possible evidence after the metal detector finds it, to determine its evidential significance. But we are still better off using the metal detector than laboriously digging up the entire site by hand.

For more information on the conceptual design of TACL tests for particular research questions, and Buddhological-philological method for its use, users should consult the ["TACL Methods Guide"](#).

TACL was originally written in [python](#), for use at the [command line](#). The full set of TACL tools is still (always) freely available in this form [at GitHub](#), and you can learn how to use that version of TACL through the [TACL documentation](#). In that world, TACL operations look like this:



```
Command Prompt
C:\Users\dg152\Documents\TACL local\GUI Roland Borsos\GUI user manual trials>tacl ngrams -v "T350 T251 db.db" "corpora/T350 T251 corpus" 2 8
2022-07-08 16:58:25,327 tacl.data_store INFO: Creating database schema, if necessary
2022-07-08 16:58:25,327 tacl.data_store INFO: Adding record for text T0251\@.txt
2022-07-08 16:58:25,327 tacl.data_store INFO: Adding n-grams (2 <= n <= 8) for T0251\@.txt
2022-07-08 16:58:25,327 tacl.data_store INFO: Adding 203 unique 2-grams
2022-07-08 16:58:25,327 tacl.data_store INFO: Adding 228 unique 3-grams
2022-07-08 16:58:25,327 tacl.data_store INFO: Adding 241 unique 4-grams
2022-07-08 16:58:25,327 tacl.data_store INFO: Adding 246 unique 5-grams
2022-07-08 16:58:25,327 tacl.data_store INFO: Adding 249 unique 6-grams
2022-07-08 16:58:25,327 tacl.data_store INFO: Adding 252 unique 7-grams
2022-07-08 16:58:25,327 tacl.data_store INFO: Adding 252 unique 8-grams
2022-07-08 16:58:25,327 tacl.data_store INFO: Adding record for text T0251\@.txt
2022-07-08 16:58:25,327 tacl.data_store INFO: Adding n-grams (2 <= n <= 8) for T0251\@.txt
2022-07-08 16:58:25,327 tacl.data_store INFO: Adding 204 unique 2-grams
2022-07-08 16:58:25,327 tacl.data_store INFO: Adding 229 unique 3-grams
2022-07-08 16:58:25,327 tacl.data_store INFO: Adding 241 unique 4-grams
2022-07-08 16:58:25,327 tacl.data_store INFO: Adding 246 unique 5-grams
2022-07-08 16:58:25,327 tacl.data_store INFO: Adding 249 unique 6-grams
2022-07-08 16:58:25,342 tacl.data_store INFO: Adding 252 unique 7-grams
2022-07-08 16:58:25,343 tacl.data_store INFO: Adding 252 unique 8-grams
2022-07-08 16:58:25,343 tacl.data_store INFO: Adding record for text T0251\@.txt
2022-07-08 16:58:25,343 tacl.data_store INFO: Adding n-grams (2 <= n <= 8) for T0251\@.txt
2022-07-08 16:58:25,343 tacl.data_store INFO: Adding 203 unique 2-grams
```

However, experience has shown that for users with typical levels of computing savvy and skills (like ourselves, pre-TACL), the command-line version of TACL is somewhat difficult to use. The TACL GUI was designed to be more user-friendly, in the hope of giving a wider range of users easier access to the tool.

We have aimed to achieve this user-friendliness not only by allowing users to interface with TACL in a manner closer to common modes of computer use, but also by careful selection of a subset of TACL functions. Some functions of the fully-fledged TACL are not available through the GUI; others have been incorporated into streamlined workflows, and are achieved as default options "under the hood", without the user having any choice. As already mentioned, in making these design decisions,

we had two types of research question in mind, which in our estimation are likely to be the most common and basic uses of TACL for real research: (1) Finding the sources of a given text in other texts, most particularly, sources that indicate a text was composed in China rather than translated (cf. e.g. Radich 2014); (2) Finding distinctive stylistic traits of a given author, translator, or translation group, with application to questions of ascription and (by implication) dating (cf. e.g. Radich and Anālayo 2017, Radich 2017a). Users who chafe at the limitations of the GUI may (eventually) like to invest the time required to "upgrade" to the command-line version.

Before we get into the specifics of running the TACL GUI, it will be useful to give a simplified overview of how TACL works (including the GUI version), and the typical workflow as we work with TACL. In this section, we will give a first overview of the operation of the tool in general terms. Later in this User's Manual, we will go through each individual step in the workflow in full detail, with hands-on exercises, to train users to actually implement each step for themselves. It is simplest to conceive of the core workflow in eight steps:

- 1) select or construct a *corpus*
- 2) generate or import a *database*
- 3) compose or select a *catalogue*
- 4) run the core test: *Intersect* or *Difference*
- 5) (optionally) concatenate two or more tests in a series: *Supplied Intersect*
- 6) post-process raw results: "*Filter/Rationalize*"
- 7) import filtered results to a tool like Excel, and sort
- 8) examine n-grams back in context, in the actual texts

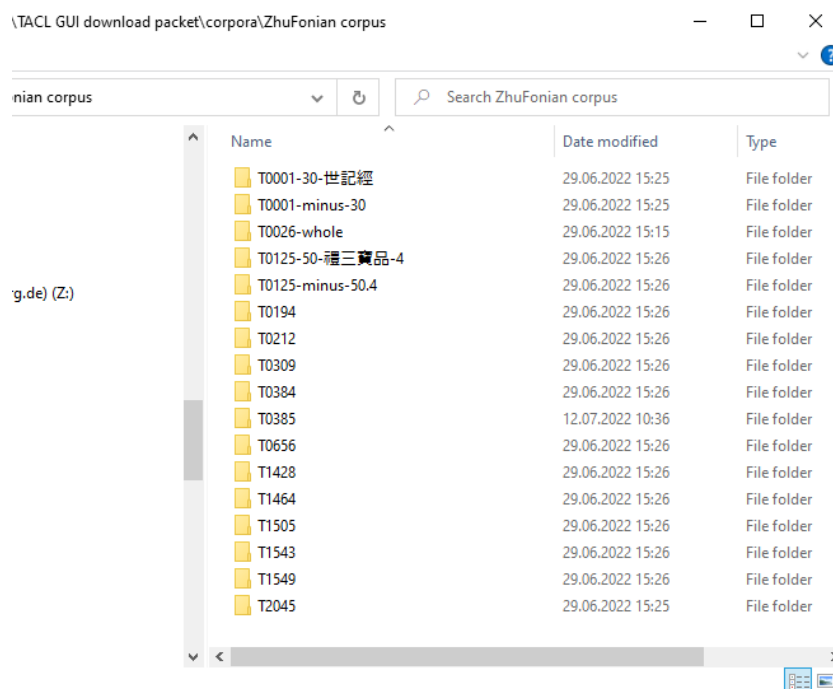
1) The corpus: TACL works with the texts of the Taishō (and other collections like the Xuzangjing/Zokuzōkyō 續藏經, etc.) as digitised by [CBETA](#)—specifically (at the time of writing), the 2019 release, as available on the [CBETA GitHub](#). TACL ultimately requires texts in text-only format. To that end, TACL pre-processes the XML in two steps: the first creates a preliminary form easier for the code to work with; and then the second process "strips" the XML tagging out of the files, to produce plain text. However, TACL GUI users do not need to worry about these pre-processing steps. To make use of the TACL GUI easier, these steps have been pre-applied to two full corpora, which users can download (see below).

Users should note that at this step, TACL uses the XML versions of the Taishō critical apparatus (the footnotes) to reconstruct text-only files representing *all witnesses* consulted by the original Taishō and/or CBETA editors in the preparation of the canon (or the CBETA digitisation). That is to say, for a given single Taishō text, e.g. the *Da zhidu lun* 大智度論/\*Mahāprajñāpāramitopadeśa T1509, the

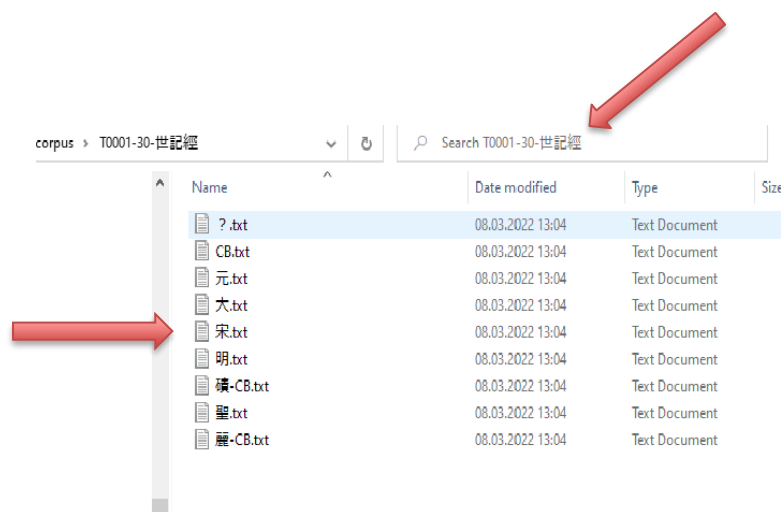
TACL corpus includes not just one version of the text, but a file *each* for the versions of the Taishō 大, Song 宋, Yuan 元, Ming 明, Kunaishō 宮, Ishidera 石, etc. This means, in effect, that TACL handles the information in all of these various witnesses—we can metaphorically imagine this scenario by thinking of it as "searching the Taishō (CBETA) footnotes". (Users should note that this whole process is of course only as reliable as the Taishō and CBETA editorial work—TACL has no direct access to other published or manuscript versions of these witnesses, and is therefore prone to any errors that the Taishō or CBETA editors might have made.)

Once the digitised files are thus transformed into the format TACL requires, the user selects or constructs a particular *corpus* to work with. It is possible to work with large corpora, such as the entire Taishō, but this can be hard work for the computer (see further below). It is also possible to work with small corpora tailored to particular research problems. We describe the simple process of creating such a bespoke corpus in the "WORKFLOW STEP 1: Corpora" section below (p. 19).

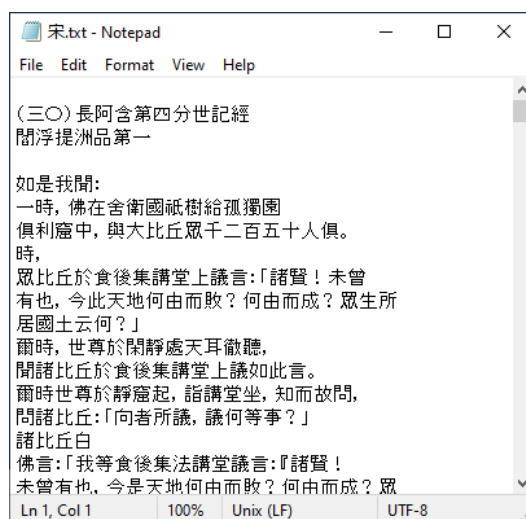
An example of such a smaller corpus is found in your "starter kit", and we can use it to show quickly what a typical corpus looks like. This is a corpus for the works of Zhu Fonian, and some associated works we have included for the purpose of our exercises below: see "corpora/ZhuFonian corpus". When we open the corpus, we see at the top level that the corpus contains subdirectories, usually one for each work in the corpus: *Dirghāgama* T1, the *Ekottarikāgama* T125, the *Udānavarga* T212 and so on. Some of the works we see here in fact have two subdirectories, because we use here the texts from the "Radich" Taishō corpus, in which these texts have been split into parts to allow for separate study—see [here](#).



When we look inside one of these subdirectories for a work—we will use as our example the first, T0001-30-世記經 (*sūtra* no. 30 in the *Dirghāgama*, the 世記經)—we see that it contains a number of text files, one for each witness:



For example, the file called "宋.txt" is what the Taishō editors called the "Song" witness to T1(30), reconstructed according to the information of the Taishō apparatus (as digitised by CBETA). If we open such a file in a text editor, we see the text-only file, as the final product of the TACL pre-processing steps described above, which further TACL processes work upon:



2) The database: This, then, is what a corpus looks like "under the hood". For any user-defined *corpus*, we then use TACL to generate a *database* containing all the *n-grams* in that corpus, and the counts for each *n-gram* in each *witness* for each *work* in the corpus.

When building the database for any given corpus, users define the size range for the *n-grams* they are interested in. We typically recommend that a range of 2- to 8-grams, based on our experience in trying to strike a balance between two concerns: on the one hand, we want to catch most items that will recur a significant number of times, which makes sure that most of what we are interested in will be in the database; on the other hand, we want to keep the database to a workable size. If the database were to include very long strings (e.g. up to 100-grams), the vast majority of what it records would be unique strings, of relatively little evidential value for most purposes; but the database would swell to immense, unusable size, and also take an unreasonably long time to generate in the first place.

To repeat, then: the *database* contains all the *n-grams* in the corpus, and the counts for each *n-gram* in each *witness* for each *work* in the corpus. This means that a typical row in the database would contain information of this sort (here presented in simplified, schematic form):

佛言, T0001-30-世記經, 宋, 5

meaning that 佛言 appears 5 times in the Song witness of T1(30).

Note that this is all the information that the database stores. Most particularly, it knows *that* a given *n-gram* occurs in a given witness of a given work, and how many times—but it does not know *where* in the text the *n-gram* occurs (it stores no information about the internal structure of the text whatsoever). Questions about how the evidence sits or works in context are the job of the human researcher, as we will see below.

All subsequent TACL tests—most typically, Intersect and Difference tests—actually work out of this database, rather than examining the corpus directly.

3) The catalogue: The last step in preparations to run a test is the preparation of a so-called *catalogue*. When we are working out of a given *corpus* and *database*, we do not necessarily want every text in the corpus to be the target of every test we run, but on the other hand, it would be inefficient to build a new database for every test; the catalogue thus allows us to specify a subset of the texts in the whole corpus as the target of a given text. We also need to tell the computer how we want to group the texts in the corpus: any Intersect or Difference test is a comparison between two

(or more) things, but without the information in the catalogue file, the only category that any text falls into is the corpus as a whole. The so-called TACL *catalogue* is a text document we use to (optionally) select a subset of all texts in the corpus, and group the texts under comparison. It is basically a list of texts, marked up to show the larger group of texts to which each belongs (see further below).

4) The core tests: We now have a *corpus* with the texts we want to analyse, a *database* containing information about the counts of n-grams in each witness of each text, and a *catalogue* identifying the groups of texts we want to compare. We then run the core TACL operations: Difference, or Intersect. The tool finds either: all n-grams (and counts) found in some text on both sides of the comparison (*Intersect*); or all n-grams (and counts) found in some text on only one side of the comparison (*Difference*). Raw results of such tests are output in .csv ("[comma separated values](#)") format; on this format, see further below).

5) Concatenate tests (optional): As already mentioned, we then have the option of "concatenating" tests in a linked series, using Supplied Intersect. This enables us, for example, to combine the results of an Intersect and a Difference test—for example, to find n-grams in BOTH text-group A and text-group B, but NOT in text-group C.

6) Post-processing raw results: The next step in our usual workflow is that we post-process the raw results produced by these tests, in order to tidy them up, and sometimes filter out only part of the results most likely to be of interest. In TACL GUI parlance, we call this step "Filter/Rationalize". The purpose of this post-processing step is to eliminate redundancy in the results, and make it easier for the human analyst to work with raw results, and efficiently find evidence relevant to the research question.

We can illustrate some of the advantages for human analysis derived from such post-processing with a simple example. As we have just seen, each database has a user-defined maximum n-gram size (string length)—usually 8-grams. At the same time, there are obviously some purposes for which we are ultimately interested in strings longer than 8-grams. For example, the following 67-gram is shared by two texts only—T150A and T735—and forms part of a pattern of evidence that shows that those two texts have a close and peculiar relation (see Nattier 2008: 51, 131; below, we will return to this example and show users how to generate this result themselves):

...盡為生死盡識何等為生死欲盡受行識為是八行識諦見至諦定為八如是為生死欲滅受行  
識何等為生死味識所為生死因緣生樂喜意如是為生死味識何等為生死...

T150A (II) 876b13-17; T735 (XVII) 537c2-6.

If we stick with results in which the maximum string length was 8, this match between T150A and T735 would be represented by 60 rows in the results file, showing the sixty distinct 8-grams that are found within this single 67-gram:

盡為生死盡識何等  
為生死盡識何等為  
生死盡識何等為生  
死盡識何等為生死  
盡識何等為生死欲  
... (etc.)

For a human analyst, this is a very fragmented and redundant way of discovering what, from our perspective, boils down in the final analysis to a single item of information.

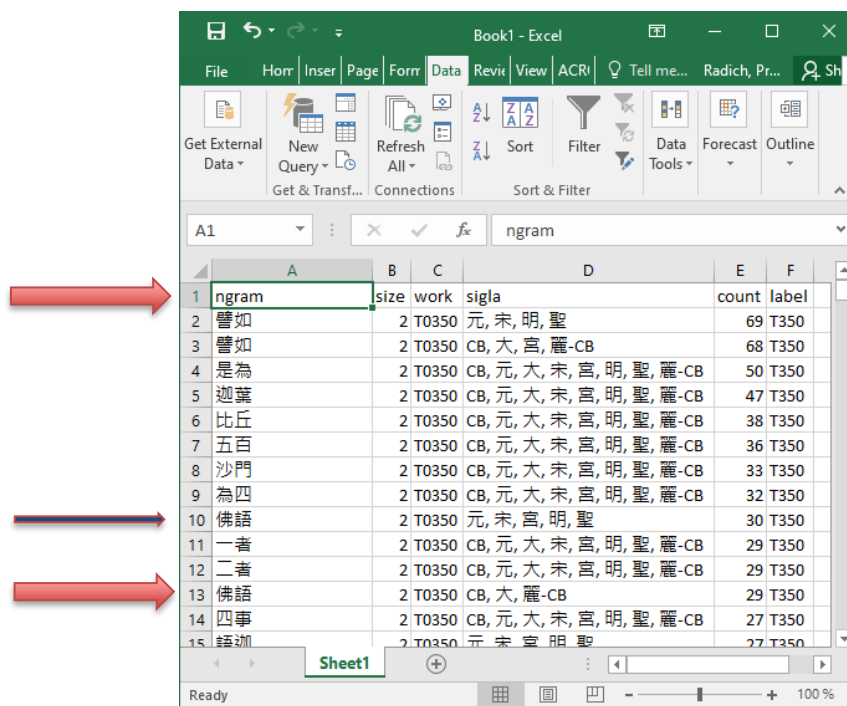
Fortunately, then, TACL can in fact return results that include such longer strings, even though they are not directly included in their complete form in a database of 2-8-grams. This work is achieved by a function we call "extend". In the TACL GUI, "extend" is included by default in the Filter/Rationalize post-processing for all results of TACL Intersect tests. This then means that the human analyst is presented with long n-grams like the above 67-gram in the post-processed results.

Alongside the work of finding such longer strings of interest, which are initially excluded from direct discovery by the parameters of the database, TACL GUI "Filter/Rationalize" post-processing also allows us to filter results by such factors as string length, hit count, or the number of works (texts) in which strings occur. These tools allow users to sift results, which can otherwise be overwhelmingly copious in quantity, to zoom in more effectively on strings likely to be of evidential value for various research questions. (See further the "TACL Methods Guide".)

To summarise thus far: in simplified form, a typical TACL test would proceed as follows. (1) The user constructs or selects a corpus, and (2) on that basis, generates a database. Next, (3) the user writes a "catalogue" defining two or more groups of text (corpora) to be compared. (4) The user then most typically runs a comparison—a Difference or an Intersect—on those texts/corpora. (5) Optionally, the user might run further tests, concatenating the raw results of two or more prior tests. Finally, (6) the user post-processes ("Filter/Rationalizes") the raw results of the test to find longer strings of interest (where relevant), or filter the results by count, distribution, etc. The work of the machine is at this point mostly done, and now the human partner in this cyborg operation takes over, to begin qualitative philological analysis of the raw results.

7) Viewing and analysing in Excel (or something similar): All basic TAFL operations—meaning mainly Difference, Intersect, and post-processing ("Filter/Rationalizing")—output raw results in a file format called .csv ("[comma separated values](#)"). Subsequently, we usually import raw results into a tool like Excel. In Excel, we can also sort results, according to protocols fitted to each test type (we also describe these sort protocols in detail below). In a sense, the work of the human researcher only really begins at this point. The human researcher needs to pore over those results to see what is of evidential value—like the archaeologist assessing the metal dug up using the metal detector.

Once we have results imported and sorted in a tool like Excel, then, users find themselves looking at something like this:

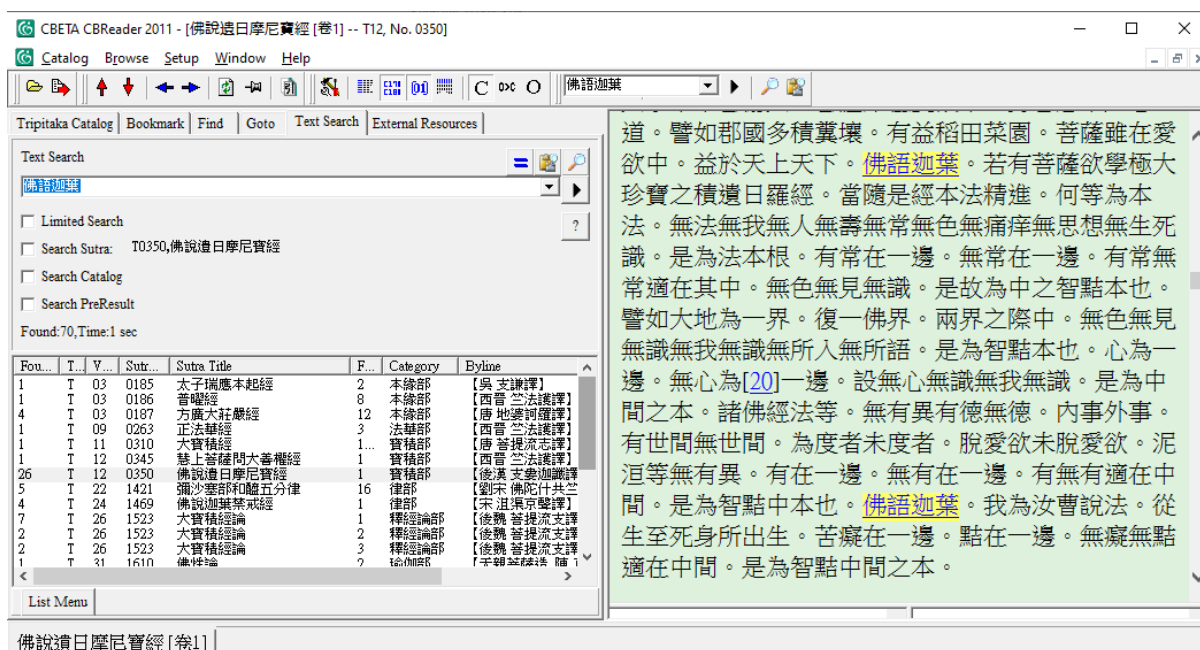


1	ngram	size	work	sigla	count	label
2	譬如	2	T0350	元, 宋, 明, 聖	69	T350
3	譬如	2	T0350	CB, 大, 宮, 麗-CB	68	T350
4	是為	2	T0350	CB, 元, 大, 宋, 宮, 明, 聖, 麗-CB	50	T350
5	迦葉	2	T0350	CB, 元, 大, 宋, 宮, 明, 聖, 麗-CB	47	T350
6	比丘	2	T0350	CB, 元, 大, 宋, 宮, 明, 聖, 麗-CB	38	T350
7	五百	2	T0350	CB, 元, 大, 宋, 宮, 明, 聖, 麗-CB	36	T350
8	沙門	2	T0350	CB, 元, 大, 宋, 宮, 明, 聖, 麗-CB	33	T350
9	為四	2	T0350	CB, 元, 大, 宋, 宮, 明, 聖, 麗-CB	32	T350
10	佛語	2	T0350	元, 宋, 宮, 明, 聖	30	T350
11	一者	2	T0350	CB, 元, 大, 宋, 宮, 明, 聖, 麗-CB	29	T350
12	二者	2	T0350	CB, 元, 大, 宋, 宮, 明, 聖, 麗-CB	29	T350
13	佛語	2	T0350	CB, 大, 麗-CB	29	T350
14	四事	2	T0350	CB, 元, 大, 宋, 宮, 明, 聖, 麗-CB	27	T350
15	語迦	2	T0350	元, 宋, 宮, 明, 聖	27	T350

The top row (Row 1) gives column labels, which tell users what they are looking at. In Column A, we see the n-gram—using the example in Row 13, 佛語. In Column B, we see the size of the n-gram (string length). In Column C, we see the work for which that row gives results, and in Column D, "sigla", we see the witnesses of the work to which the result shown applies, indicated using the abbreviations of the Taishō/CBETA apparatus. In Column E, we see the count for the n-gram in question in all the witnesses listed (in Column D) for the work in question (in Column C). In Column F, we see the "label" that we used in the TAFL *catalogue* to group the text into a group for comparison. Thus, Row 13 here, for example, shows us that 佛語, a 2-gram, appears 29 times in witnesses CB, 大, and 麗-CB of T0350 (Lokakṣema's *Kāśyapa-parivarta*) (note that in Row 10, marked by the smaller blue arrow, we see that the same n-gram appears 30 times in other witnesses of the same text).



8) Examining results in context: The next step in human, qualitative philological analysis of the raw results is typically that one needs to take n-grams in the raw results that appear interesting, and examine them back in context, to figure out what they might mean for the research question. Remember, once more, that TACL does not "know" *where* in the texts n-grams occur, nor what they mean. Analysis of those aspects of the n-grams TACL discovers is the job of the human. We most typically do this work through the use of a tool like the CBReader, e.g.:



Below, we will describe in detail each of these steps in the use of the TACL GUI, in this order, with hands-on exercises through which you can learn to do it yourself. But first, a couple more preliminaries.

### **TACL-related methodology**

In order to use TACL for research in the study of Chinese Buddhist texts, users need two skillsets: (1) You must be able to drive TACL (via the GUI, or at the command line); (2) You also require a clear grasp of the conceptual design for a rigorous analysis using data found by TACL, and the work processes by which the human user works through results to select and apply evidence to test hypotheses and construct arguments.

This User Manual introduces only (1) technical use of the TACL GUI. Guidance on TACL-related method (2) can be found in the "[TACL Methods Guide](#)". It should also be helpful to read already published research based on use of TACL (such studies are listed in the Bibliography at the end of this manual, and PDFs are included in the "starter kit").

Users who have questions about the design of research projects using TACL are very welcome to [email us](#).

### **The TACL GUI**

As mentioned above, the TACL GUI is a simplified version of TACL (it has only a subset of the full range of functions available in the command-line version). The GUI has been designed to make use of TACL as easy as possible, for work on certain common problems.

Users who have technical problems with the TACL GUI, or discover bugs, can report them on the [TACL GUI GitHub](#), or [contact us](#) by email.

### **GUI download**

The TACL GUI can be downloaded in ready-to-run format from the [TACL GUI GitHub](#). No installation is necessary. Just put TACL-GUI.exe in a working directory where you plan to do your TACL work. (For the purposes of the exercises in this User's Manual, you can leave it for the time being in the unzipped "starter kit" directory.) The programme is ready to launch at a click. Before you launch for the first time, however, please read further.

A copy of the TACL GUI is also included in the "TACL GUI starter kit", available for download [here](#). The "starter kit" also includes materials for the various examples we present in this User's Manual, which are intended as hands-on exercises that you can work through as you read, to familiarize

yourself with the operation of the GUI. Note that the GUI in the starter kit is for Windows only. Mac and Linux users need to download the GUI directly from the [TACL GUI GitHub](#). All the screenshots and instructions in the Manual are also based on the Windows version. There should be no major difference between the two versions, aside from different default layouts inherited from the OS.

With these preliminaries in hand, we now return to the eight-step workflow that we introduced in general terms above. In the remainder of this Manual, we will work through the steps in this workflow in detail, proceeding in the same order as in the general description above. The steps in the workflow, and the operation of the GUI, will be illustrated with hands-on exercises. Those exercises will use the following real examples, though in somewhat simplified form (the examples are described in more detail below; they are listed here in order of appearance):

EXAMPLE 1: The Zhu Fonian corpus, with a particular focus on the question of who translated the *\*Ekottarikāgama* T125.

EXAMPLE 2: Shared text demonstrating a close intertextual relation between the *Qi chu san guan jing* 七處三觀經 T150A and the *Si yuan jing* 四願經 T735.

EXAMPLE 3: Shared text demonstrating a close intertextual relation between *\*Ekottarikāgama* 48.3 and the *Mile xiasheng jing* 彌勒下生經 T453.

EXAMPLE 4: Phraseology distinguishing the style of An Xuan and Yan Fotiao from that of Dharmarakṣa, as seen in their respective translations of the *Ugra-paripṛcchā*.

### **WORKFLOW STEP 1: Corpora**

As described above, TACL operations require a *corpus* to work on. TACL works with the [texts of the Buddhist canon as digitised by CBETA](#). However, as described above, TACL requires that these XML files be put through a couple of steps of preparation and modification before TACL can process them. To simplify matters for GUI users, we have left those preparatory operations out of the GUI functionality. Instead, we provide prepared CBETA corpora for download at [Zenodo](#).

Users can choose between two large corpora (click on the link for each): (a) A corpus comprising the [Taishō and Xuzangjing/Zokuzōkyō collections](#), with the content structured exactly as the Taishō/CBETA editors prepared them (DOI: 10.5281/zenodo.7751187). (b) A modified "[Radich Taishō corpus](#)" (DOI: 10.5281/zenodo.7823963), in which some texts or collections have been split into smaller constituent parts by Radich, in order to reflect more accurately some aspects of the structure of the texts/collections or their history. We will refer to this second corpus here as the "Radich corpus"; [this document](#) describes it, and the rationale for points on which it differs from the Taishō/CBETA.

Once users have either the T-X corpus, or the “Radich corpus”, they have two further choices about how they use one or the other as the basis for their own work, and especially, the selection or generation of a TACL database (see further below).

(1) Users can work directly from one of these two large corpora.

(2) However, for many restricted purposes, users may want to use only a smaller subset of the full Taishō (or some other CBETA corpus). In such cases, it can be far more efficient for users to create their own smaller corpus.

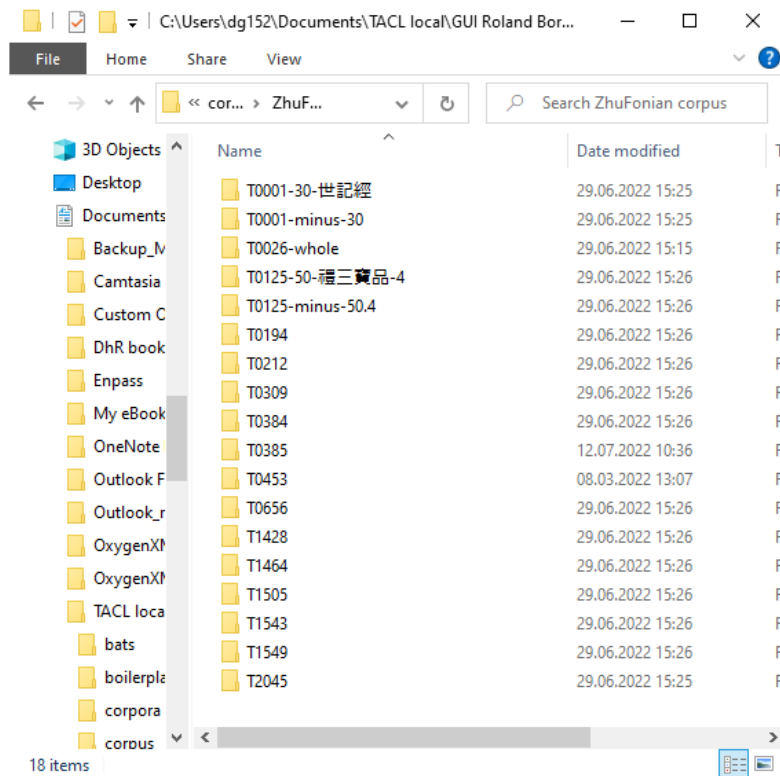
Smaller tailor-made corpora of this sort should be created by moving files from the T-X corpus or the “Radich corpus” to a separate sub-directory created for the new sub-corpus. Users can then generate a custom-made database on the basis of the tailor-made corpus (see below). For an example of such a custom, task-specific corpus, see "corpora/ZhuFonian corpus" in your "starter kit".

Two main considerations should guide your choice between these two alternatives. First, you will need to consider your research goals, and the corpus (and database) required to address those goals. Second, however, you will also need to consider the practical difficulties of importing or building a large database, which can be considerable (see below).

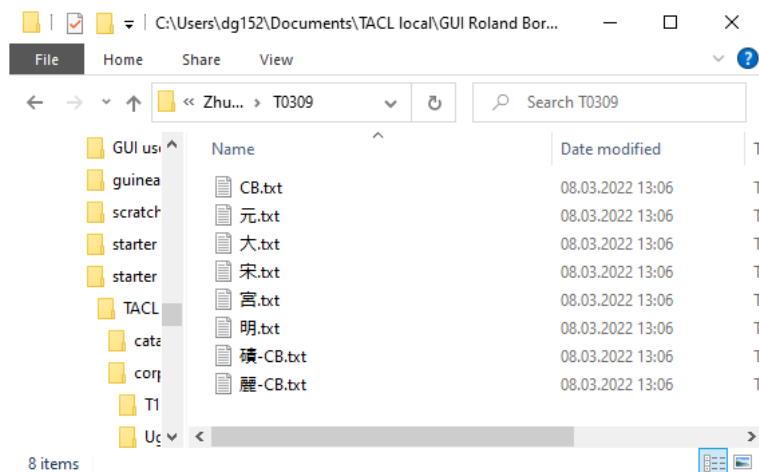
Either way, the TACL GUI requires at least one corpus to work on in order to operate. You should therefore have a corpus ready when you first launch the GUI—either a corpus of your own selection and making, or one of the “standard” TACL download corpora.

It is important to note that TACL and the GUI require that corpora have a precise structure, already described more briefly above. This information is particularly relevant for users who want to add texts or corpora from sources other than CBETA.

The corpus as a whole must be a single directory (folder). The corpus directory contains *subdirectories*, one for each *work* in the corpus. The filename for each of these directories must be exactly the identifier used for the work in TACL *catalogues*. The *subdirectory* for each work contains one or more text files (with the .txt file extension), each representing a single *witness* of the work. Examples can be seen in the "corpora" directory of your "starter kit". For example, the "ZhuFonian corpus" directory looks like this:



Each subdirectory represents a single work: for example, "T0309" is the subdirectory for the *Shi zhu duan jie jing* 十住斷結經 T309, and "T0656" is the subdirectory for the *Pusa yingluo jing* 菩薩瓔珞經 T656. The subdirectory "T0309" looks like this:



Here, "元.txt" is the text file containing the witness to T309 from the edition of the canon called "Yuan" by the Taishō editors, as documented by the Taishō apparatus, "大.txt" contains the witness of the Taishō base text, and so on. (The big corpora we have made available for download are also structured in this manner, as you can confirm for yourself by looking directly at the files.)

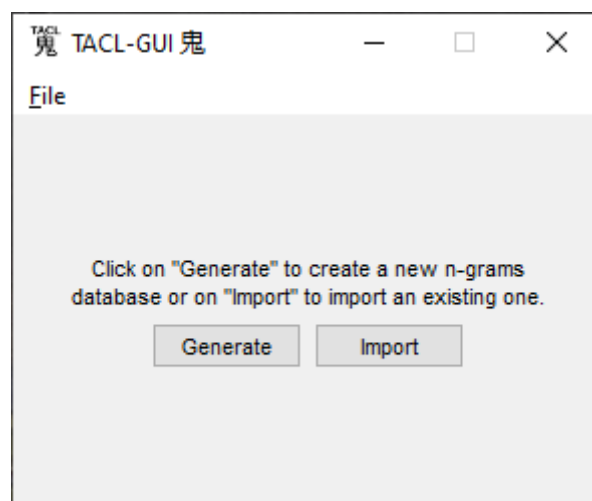
It is important for users also to note that work identifiers (subdirectory names) and filenames of witness .txt files should not contain spaces or special characters. Using spaces and special characters can create complications at later steps of the process.

The TACL GUI can only process corpora structured in subdirectories containing text files, in this manner. It is particularly important to bear this point in mind if you wish to manually add works to your corpus that are not represented in the CBETA digitisations. You must then create a subdirectory for each work, and within that subdirectory, a text file for each witness (even where you only have one witness).

In principle, it is possible to use TACL with corpora other than the files digitised by CBETA (e.g. users might be interested in working with texts from the Daozang or the *Si ku quan shu*, or digitisations of texts attested only in manuscript form). However, any such additions to the TACL corpus must be structured in the format just described. Another proviso is that texts must be digitised and tokenised in a form that TACL recognises. We recommend that users who are interested in such extensions of their corpora [contact us](#).

### **WORKFLOW STEP 2: Databases**

As described in broad conceptual terms above, TACL works with a database of n-grams and counts. This is the real basis for its analyses. TACL (and the GUI) thus requires a database of all the n-grams contained in your target corpus. When you launch the TACL GUI for the first time, you will therefore first see a dialogue allowing you to generate or import a database.



As you see, you have the choice of generating your own database within the GUI, or importing one from another source.

The main reason for this choice is that for large corpora, like the T-X corpus or the “Radich corpus”, the full database will be very large—for the “Radich corpus”, for instance, a database of 2-8 grams is about 170 GB. Unfortunately, this means that one way or another, it is a time-consuming process getting a database ready for use with the GUI.

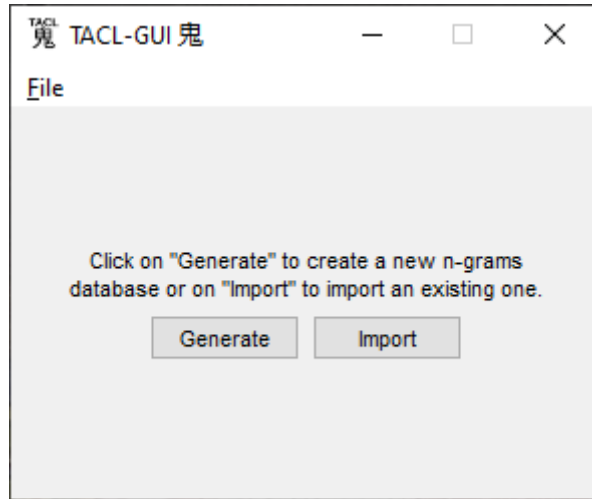
Running on a relatively powerful computer (64 GB Ram, i7 processors, SSD), the GUI takes about six hours to generate a full database for the “Radich corpus”.

Alternatively, users can download from Zenodo a [database for the full "Radich Taishō corpus](#) (the corpus described above) (DOI: 10.5281/zenodo.7824781). However, the compressed download file is about 47 GB, and so download could take a long time if your Internet connection is slow. Once the file has been downloaded, further, it must be unzipped and readied for use, which also takes even a powerful computer quite a long time, but after that, importing it into the GUI takes very little time and does not require much from the computer. For these reasons, downloading this full database, rather than building it from scratch locally, is probably the better option for users with limited computing power at their disposal. We describe download more fully below.

For users wanting to use a full corpus and database, these logistics mean that the TACL GUI requires the investment of considerable setup time. Generally speaking, however, this setup should be a one-time operation, after which regular use of the TACL GUI is somewhat more streamlined. Users wanting to use a full corpus and database should thus weigh up whether to generate their database locally, or download and import it, depending upon such factors as the power of their computer, available memory space, and Internet connection. We describe both processes in more detail below.

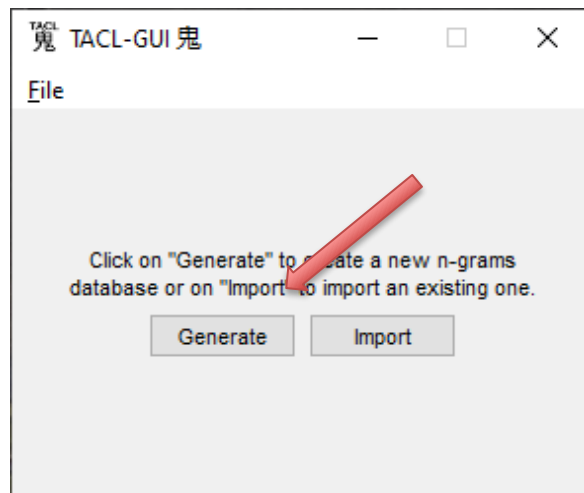
### **Generating or importing a database in(to) the GUI**

Once again, then, when you launch the TACL GUI for the first time, you first see a dialogue allowing you to generate or import a database.

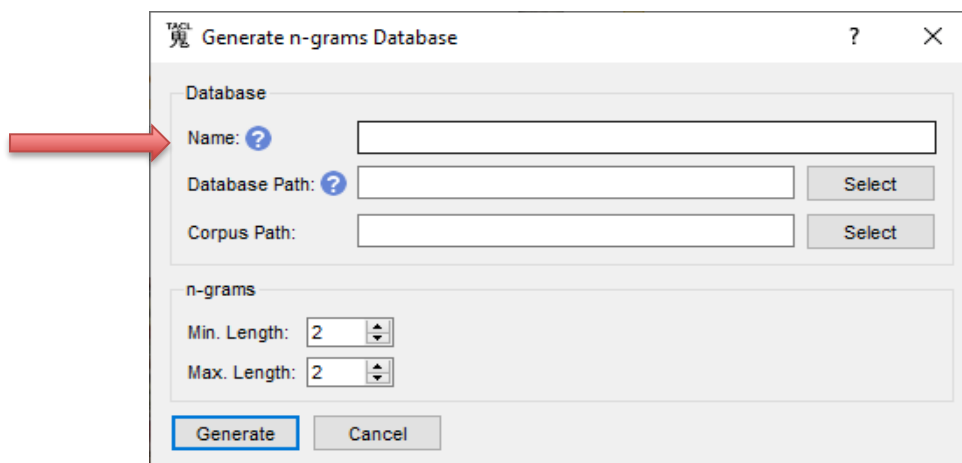


### (1) Generate a database

If you are using a corpus of your own selection and making, you now need to generate a database on the basis of that corpus. At the initial screen,

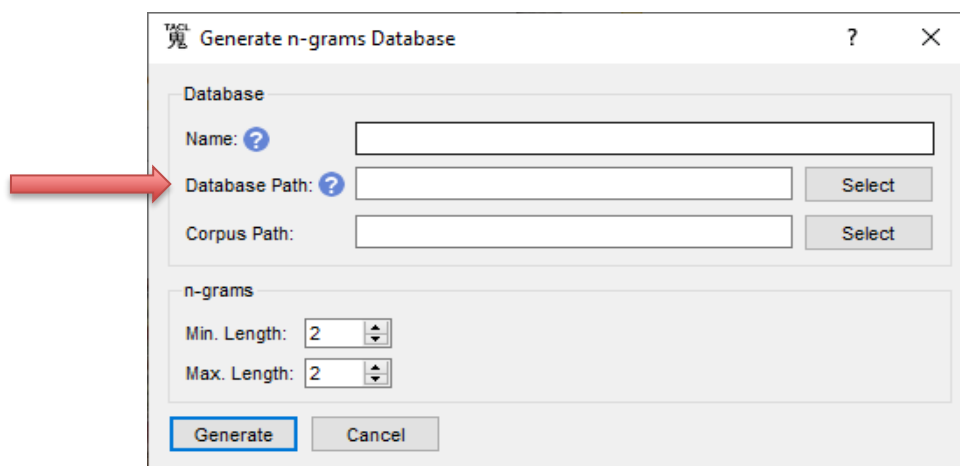


Click "Generate". You get this screen:

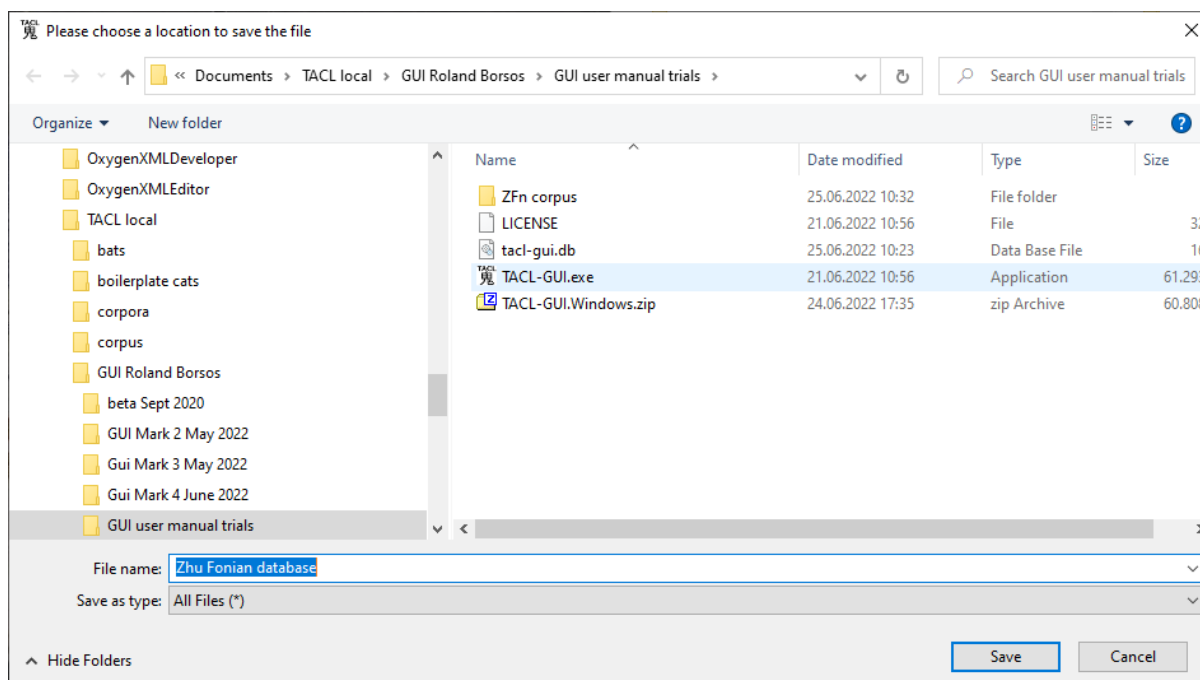




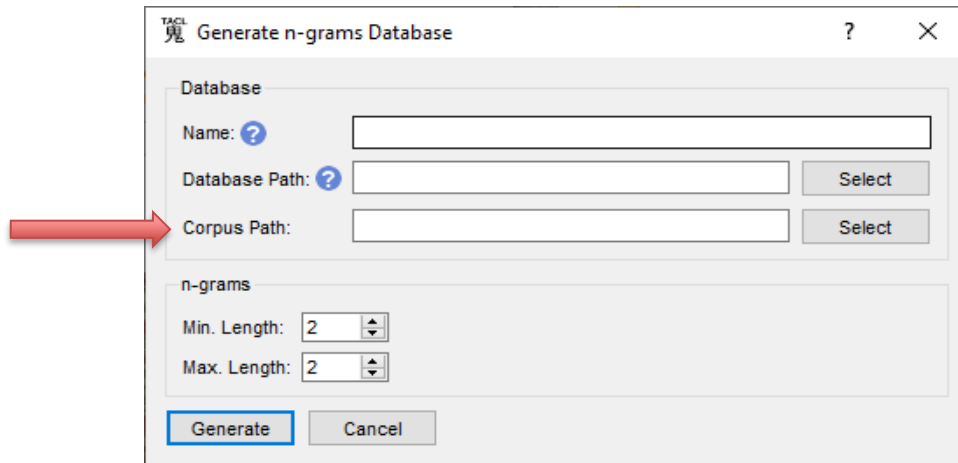
In the “Name” field, type a name for your database. We will here use as our example some smaller-scale tests on texts by Zhu Fonian (EXAMPLE 1). For that purpose, your database might be called e.g., “Zhu Fonian database”.



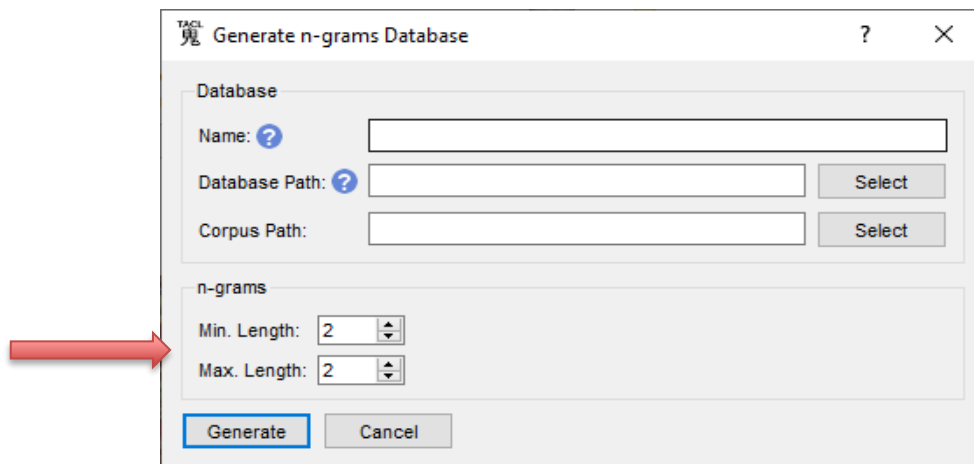
In the “Database Path” field, enter a path leading to the directory where you want to work on your TACL analysis. The simplest way to do this is to click “Select” and navigate to the directory in question. You can also directly type a path. Note that the “Database path” must include a filename at the end. The simplest option here is just to repeat the database name you just selected in the “Filename” field, as here:



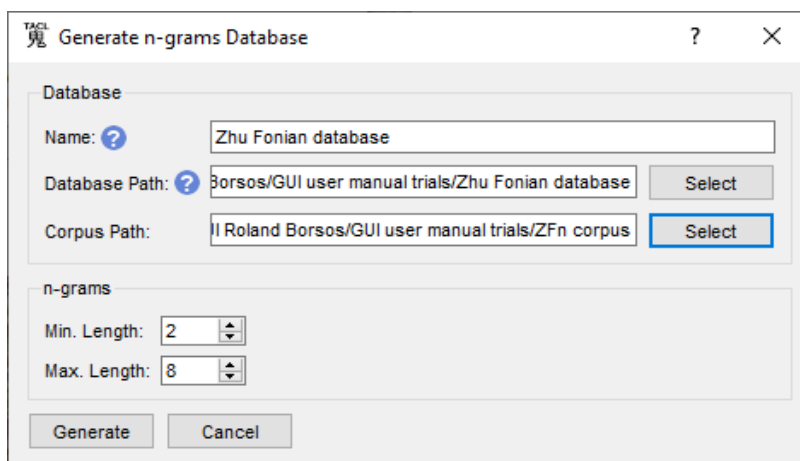
Returning to the “Generate n-grams database” screen (already shown above), you next select the corpus that you want to use as a basis for your database ("Corpus Path"). Again, click “Select”, and navigate.



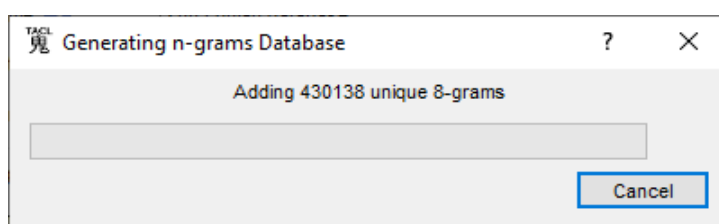
Finally, you are required to stipulate the minimum and maximum length of the n-grams that will be incorporated in your database. As explained above, we recommend min: 2, max: 8.



Your screen should now look something like this:

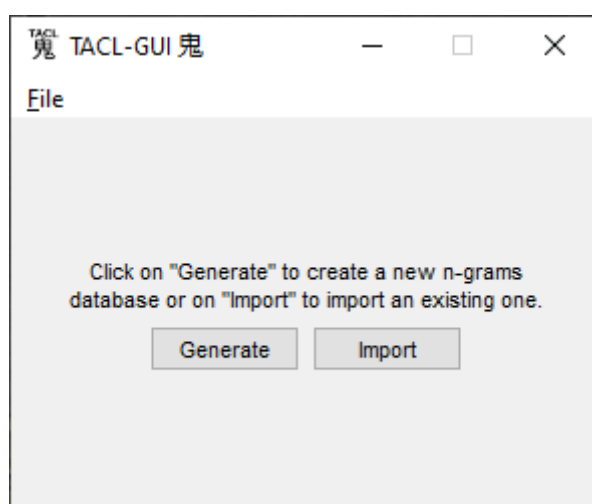


Click “Generate”. You will see a progress screen as the database is being generated:



Depending on the size of your corpus, database generation can take anything from a minute or two, to many hours or even a couple of days.

As already mentioned above, however, the first screen of the GUI gives us two options. Not only can we generate a database, we can also import one:



Alternatively, then, you might want to import an already existing database, such as the database for the "Radich Taishō corpus" as discussed above. For this purpose, we will first outline the download

process, and then describe how you can then import a database into the GUI from an external source.

(2a) Download the full database for the “Radich corpus” version of the Taishō

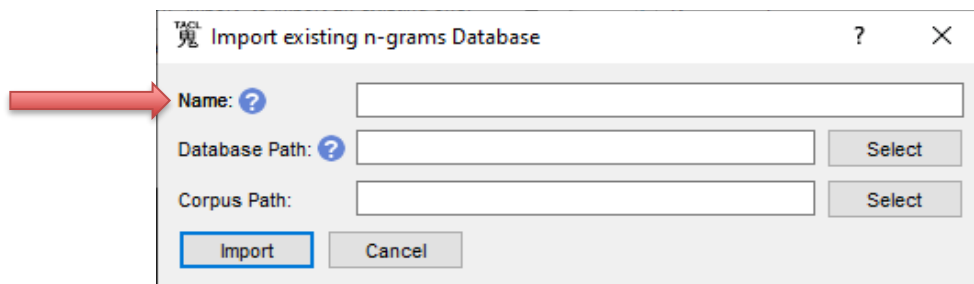
It can be time-consuming and hard work for some computers to generate a database for a full Taishō corpus. For that reason, we have made a (zipped) full database for the “Radich Taishō corpus” available on [Zenodo](#) (DOI: 10.5281/zenodo.7824781). Click [here](#) to access this download. Download size is still considerable (approx. 47 GB), and download could therefore take quite some time.

Once your download is complete, save or move the zip file to a directory you will use to do your TAACL work, and unzip the database using an appropriate utility like [7zip](#). Note again that because the file is very large, unzipping could also take quite some time (up to a few hours).

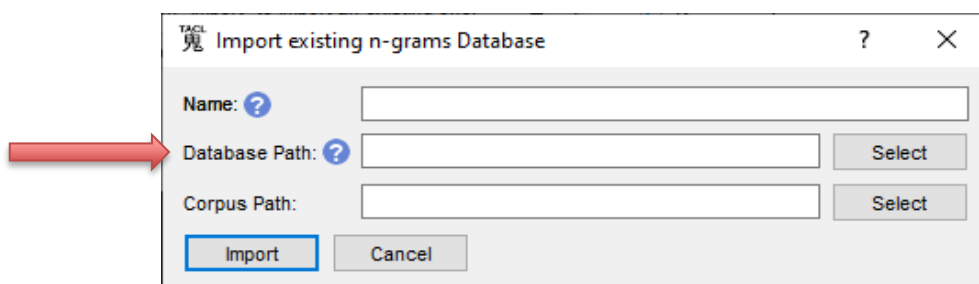
Once unzipped, this "full Taishō database" is approx. 170 GB. You should ensure you have sufficient free disk space before attempting the download and unzip.

(2b) Import a database

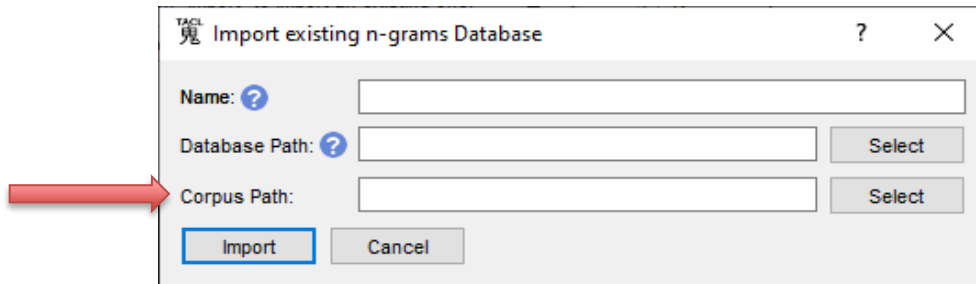
Once again, then, at the initial screen, you are presented with a choice between generating a database, and importing one (see above). If you click “Import”, you get a screen like this:



In the “Name” field, type a name for your database (e.g. “full Taishō database”).

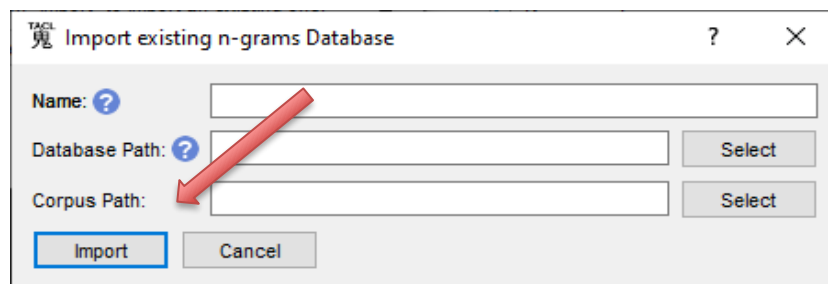


In the “Database Path” field, enter a path leading to the database you want to work from. The simplest way to do this is to click “Select” and navigate to the database file you want to import. You can also directly type a path.



In the “Corpus Path” field, enter the path leading to the directory containing the corpus associated with the database you are importing.

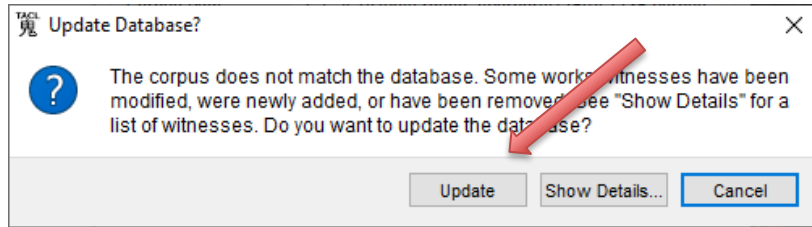
Click “Import”.



The TACL GUI will check at this point to ensure that your database and corpus match. (They must match for tests to run.) If the database you want to import is large, the process of checking the database against the corpus can take a while. You should see a sort of "progress bar" running. For technical reasons, it is not possible to display a real progress bar here—what you see is really just a placeholder graphic to assure you something is happening. Please be patient.

If database and corpus do not match (e.g. if the corpus has been modified since the database was generated from it), you will see an error message:

"The corpus does not match the database. Some works/witnesses have been modified or were newly added (see details for more information). Do you want to update the database?"

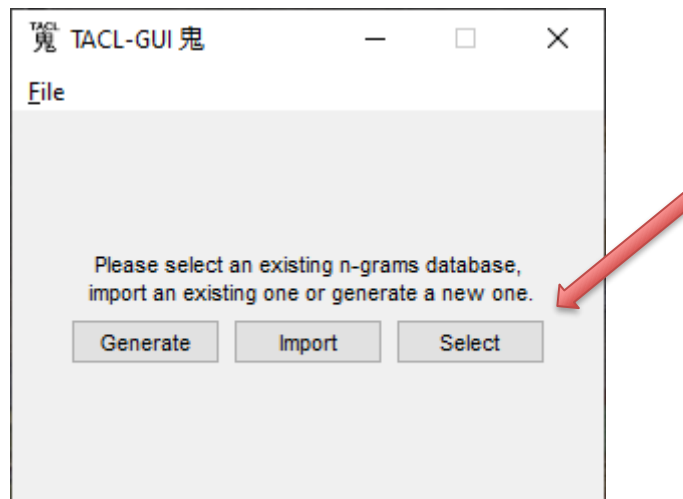


Generally speaking, you should select “update” if you see this message. The update process is a modified version of the original process to generate a database (see below), with the adjustment that the code skips parts of the corpus that have not changed. If your corpus is large, this process could take quite some time.

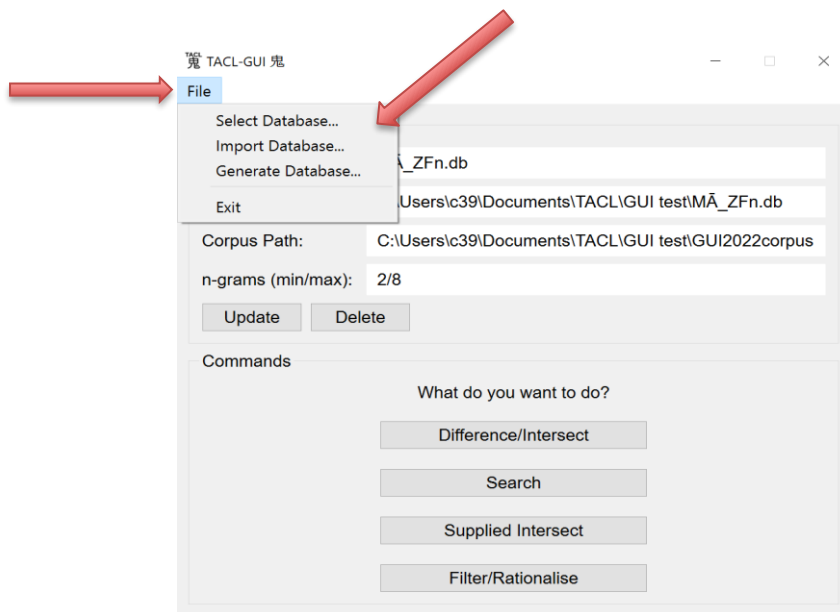
### (3) Select a database

Once you have generated or imported at least two databases, you can simply select the database of your choice for use in subsequent TACL GUI sessions. You achieve this in one of two ways:

At launch (the initial screen again), click "Select":

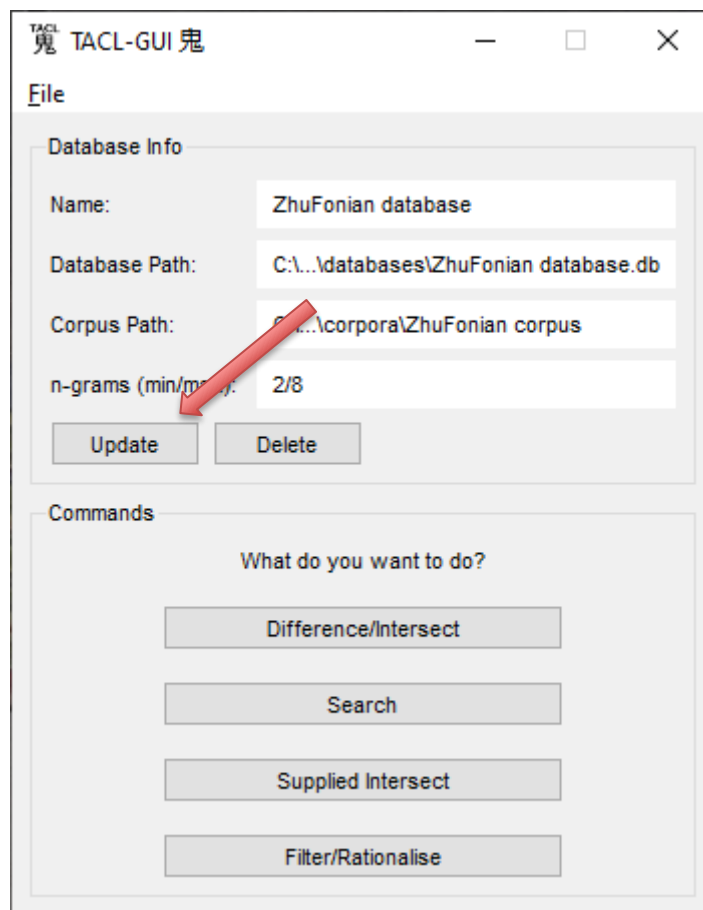


Alternatively, when already working within the GUI, you can switch databases at any time without relaunching by going to the “File” menu. In the same place, you also have the option of generating or importing a new database:

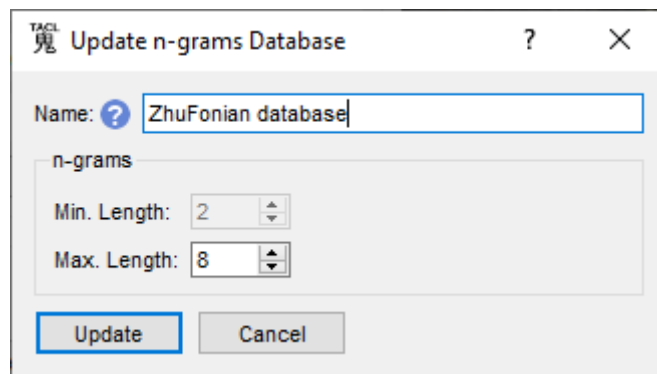


#### (4) Update a database

If you have made modifications to the corpus, you can also initiate an update yourself, without waiting for the GUI to prompt you. To initiate an update, go to the main menu, and click on the “Update” button (in the “Database Info” section).



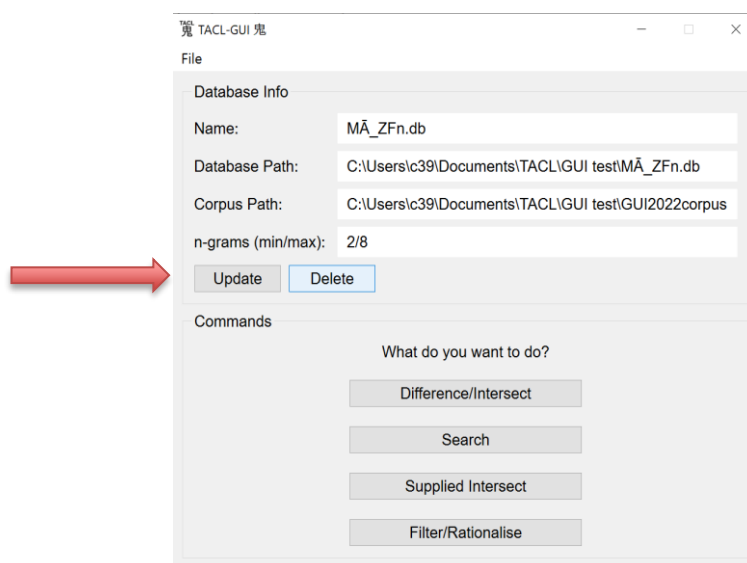
You will then see the following "Update n-grams Database" window. Here, if you wish, you can also re-adjust the lengths of the n-grams (but in principle, we recommend you leave those variables unchanged). Note, however, that there is no option here to inform the GUI if you have moved the database to a different location—in this dialog window, the path to the corpus remains the same as the one you gave when you first generated/imported the database (see below for more about changing the paths for corpus and database).



Simply click "Update" to implement the update of your database.

#### (5) Delete a database

After you select a database, you always have the option of either updating it (if, for example, your corpus has changed), or deleting it. See the "Update" and "Delete" buttons on the main TACL GUI menu screen:



Users must note that even if a database is deleted through the GUI, it is truly deleted, i.e. gone for good. If you want a deleted database back, you will either have to regenerate it, or reimport it.



“One use of one database at one time”: Note that TACL is only capable of running one operation from a given database at a given time. However, with this constraint, it should also be possible to run multiple processes in parallel on multiple different versions of the GUI—so long as each process is working out of a different database.

### **WORKFLOW STEP 3: “Catalogues”**

Use of TACL for the core functions (most typically, Difference and Intersect) requires that the user input a “catalogue”. TACL catalogues consist of two types of element: (1) lists of works, using the exact identifiers by which those works are identified in the corpus (i.e. the name of the directory containing subdirectories for each work in the corpus, e.g. "T0002", where each such subdirectory contains a .txt file for every witness of the work); and (2) “labels”, which are TACL’s way of identifying groups to which the texts belong. Users define labels directly in the catalogue files they create. The form of the labels is up to the user (we recommend using labels that will make clear sense to your future self). The only rule is that labels may not include spaces (so “ZhuFonian” is OK, but “Zhu Fonian” is not).

Here is an artificially simple example:

```
T0002 CORPUS1
T0003 CORPUS2
```

Here, the works (texts) are "T0002" and "T0003". The "labels" are "CORPUS1" and "CORPUS2". This TACL “catalogue” tells the code that we treat these texts as belonging to two different groups for the purposes of analysis, CORPUS1 and CORPUS2. Each “corpus” contains exactly one text. Using such a catalogue, we can run a comparison—an Intersect or a Difference—between these two corpora.

For the core TACL tests, Difference and Intersect, catalogues must contain at least two labels (for the sake of simplicity, we will assume two labels in all our examples here). The two labels represent the two sub-corpora you intend to compare; each text you wish to analyse needs to carry one of the two labels. If a text does not have a label in the catalogue, it will not be included in the operation.

Here is a slightly more complex real example (the catalogue we will use for EXAMPLE 1):

```
T0001-30-世記經 ZhuFonian
T0001-minus-30 ZhuFonian
T0125-50-禮三寶品-4 ZhuFonian
T0125-minus-50.4 ZhuFonian
```

T0194 ZhuFonian  
T0212 ZhuFonian  
T0309 ZhuFonian  
T0384 ZhuFonian  
T0385 ZhuFonian  
T0656 ZhuFonian  
T1428 ZhuFonian  
T1464 ZhuFonian  
T1505 ZhuFonian  
T1543 ZhuFonian  
T1549 ZhuFonian  
T2045 ZhuFonian

T0026-whole Saṅghadeva

Here, the works (texts) are T0001-30-世記經, T0001-minus-30, T0125-50-禮三寶品-4, T0125-minus-50.4, T0194, and so on. (These somewhat irregular identifiers for the works are products of the restructuring achieved in the "[Radich](#)" corpus, which, once again, is described [here](#).) The "labels" are "ZhuFonian" and "Saṅghadeva". This catalogue thus identifies the first list of texts as a single group (the works of Zhu Fonian, which we have indicated with label “ZhuFonian”); and it then identifies a single text, the *Madhyamāgama* T26, as a second, contrasting group (the entire corpus attributable with confidence to Saṅghadeva). With this catalogue, we can run TACL comparisons to find possible similarities or differences between Saṅghadeva = the *Madhyamāgama*, and works translated by Zhu Fonian.

TACL requires that catalogue files be formatted as plain text files (saved with the .txt file extension). Users can compile and manipulate catalogues in a plain text editor. We recommend [Notepad++](#) for this purpose.

Note that if you are working with a large corpus, catalogue files can become quite long. For one version of the full Taishō corpus that we used in preparing this manual, for example, the catalogue is 3,880 lines long. This naturally means it would be very laborious to construct such large catalogue files by manually typing every line. We have therefore made available “boilerplate” base catalogues of both of the large corpora we provide for download (the basic CBETA T-X corpus, and Radich’s modified T corpus): see

- "catalogues/T-X base cat.txt"
- "catalogues/Radich Taisho base cat.txt"

in the "catalogue" directory of your TACL GUI starter kit; see also [here](#)). We also recommend strongly that users familiarise themselves with simple applications of [regular expressions](#) (regex)<sup>1</sup> for find-and-replace operations, to handle the modification and especially labelling of works in such large catalogues as efficiently as possible.

Concretely, then, once you have a corpus, you proceed as follows to build a catalogue for analysis of some or all of the works it contains. Working in a .txt file, you first construct a list of the identifiers (names of the work subdirectories) for all the works in the corpus that you need for your test. You can do this one of two ways: either (a) type up a list of the subdirectory names/identifiers (only really practicable if the corpus is small), or (b) edit a ready-made ("boilerplate") catalogue (the better option when your corpus is large). Next, add a label to the end of the line for each work name, ensuring that there are at least two labels in total. Again, it helps to remember that TACL will ignore any work identifier that does not have a label. This means that if you start with one of the large "boilerplate" base catalogues mentioned above, you can often more efficiently select smaller groups of texts simply by adding labels, leaving out all the texts you wish to disregard. By contrast, deleting all the work identifiers for works that do not interest you is likely to be far more laborious, and a waste of your time.

### **Modifying the corpus, database and/or catalogue(s)**

In working with TACL, we sometimes begin with a certain corpus and database, and then realise that we want to modify the corpus (e.g. by adding or deleting works), which requires a corresponding modification of database. For example, we might have a small corpus and database for work on a specific translation group, and discover that an additional work, excluded from the initial corpus, is relevant.

Before users modify corpora, however, they should consider that it is also possible to manage the scope of TACL operations using catalogues (see above). In some cases, this is the best solution, rather than modifying the corpus or database directly. This is especially relevant when one realises that one has been *including* a work that one subsequently wishes to *exclude*. It is often more efficient in such cases to leave the database as it is, and simply omit the work to be excluded from the catalogue.

For example, let us imagine that in working with the "Zhu Fonian corpus" already introduced above, we decide that one work—our example will be T2045—is not, in fact, a work of Zhu Fonian, and we want to adjust our tests accordingly. Rather than deleting T2045 from the corpus and

---

<sup>1</sup> Another useful resource: create and test regex search and replace operations in real time at <https://regexpr.com/>.

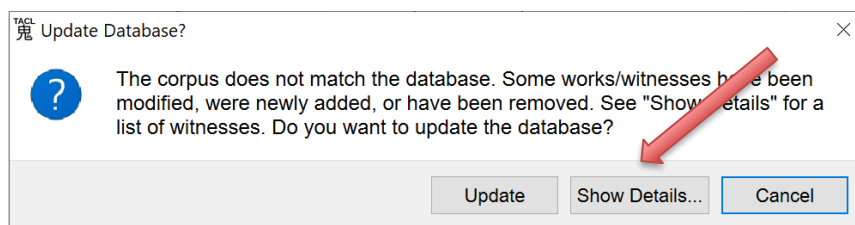
updating the database accordingly, we could simply leave T2045 without a label in the catalogue file, as follows:

```
T0001-30-世記經 ZhuFonian
T0001-minus-30 ZhuFonian
T0125-50-禮三寶品-4 ZhuFonian
T0125-minus-50.4 ZhuFonian
T0194 ZhuFonian
T0212 ZhuFonian
T0309 ZhuFonian
T0384 ZhuFonian
T0385 ZhuFonian
T0656 ZhuFonian
T1428 ZhuFonian
T1464 ZhuFonian
T1505 ZhuFonian
T1543 ZhuFonian
T1549 ZhuFonian
T2045

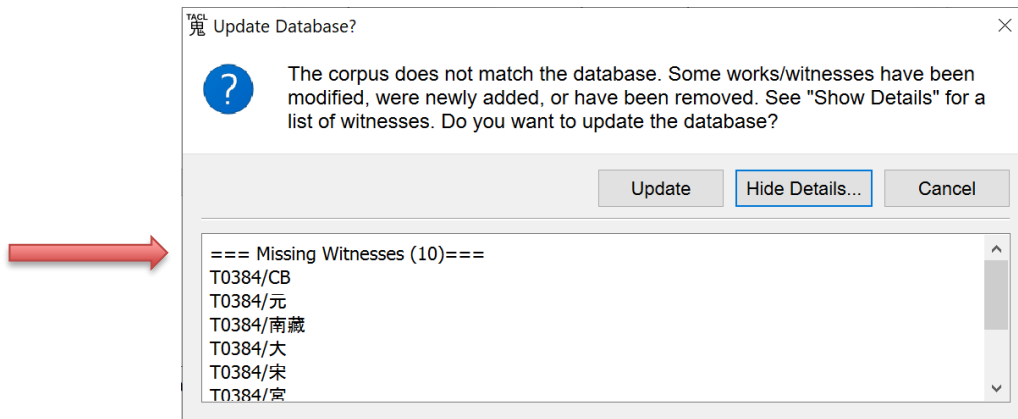
T0026-whole Saṅghadeva
```

Works without labels are ignored by TACL. Alternatively, you can delete the entire line(s) for the work(s) in question from the catalogue file. In either case, T2045 will be left out of any tests run using this catalogue. This move might be simpler and less time-consuming than regenerating the database, especially if the database is large.

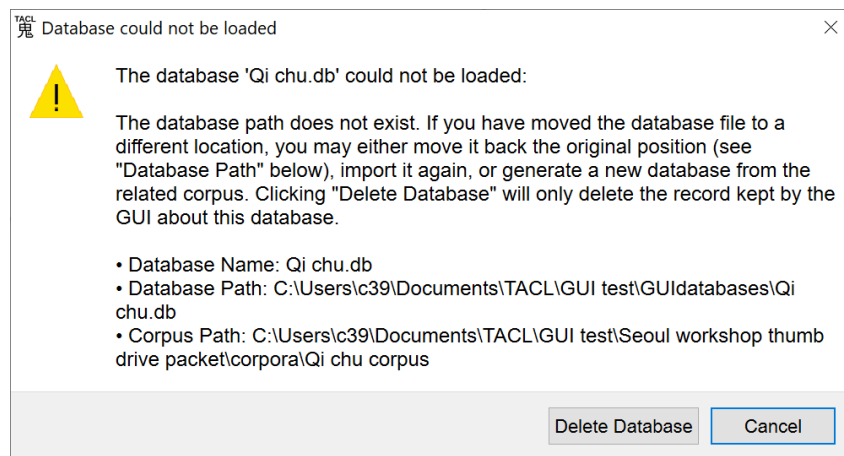
However, if we realise that a work of interest is *missing* from our corpus and database, we have no choice but to modify the corpus and update the database accordingly. In such a case, we might add the new work to the corpus. If you do modify a corpus directly, the next time you launch GUI or attempt a TACL operation, you will see this prompt to update the database:



If you are unsure about what causes the mismatch, click “Show Details”, and you will see a list of the works that cause the discrepancy:



Note also that if you move a database or corpus to a different location, or rename either the database or the corpus, the GUI will throw an error and complain that the database path does not exist.



You will then have these options:

- 1) “Delete database” (which in this case, in effect, means only deleting the old database path known to the GUI);
- 2) Move the database back to the original location or revert to the original filename;
- 3) Re-import the database from the new location/with the new filename; or
- 4) Generate the database again from the corresponding corpus.

#### **WORKFLOW STEP 4: Running core TACL tests**

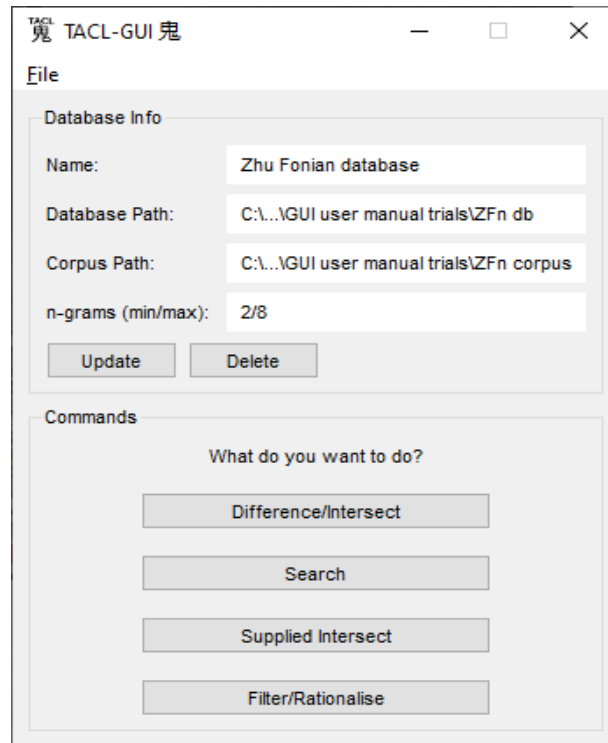
Once you have generated or imported at least one database, and you know what a catalogue file is, you are ready to run actual TACL tests.

*Note:* For all of the following functions within the GUI, you see a question mark icon:



The user can hover the mouse over this icon to get additional information about the function in question (please try it).

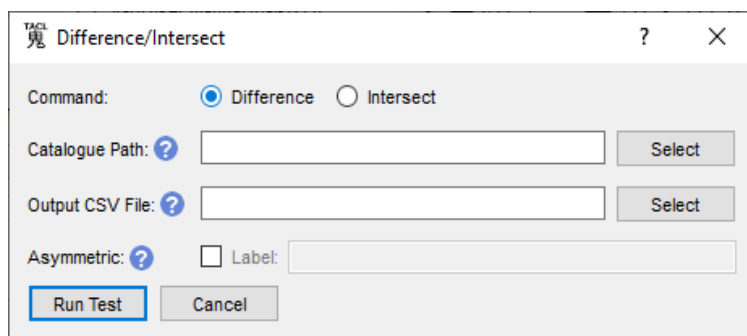
Immediately after the generate/import database process, you will see the TACL GUI "main menu" screen:



On second and subsequent uses of the TACL GUI, if you have already generated or imported at least one database and later relaunch, you will see this screen immediately at launch (unless the GUI detects a mismatch between your database and corpus—see again above).

The most basic TACL tests are “Difference” and “Intersect”. We will examine these first.

Click on Difference/Intersect, and you will next see this screen:



## Intersect

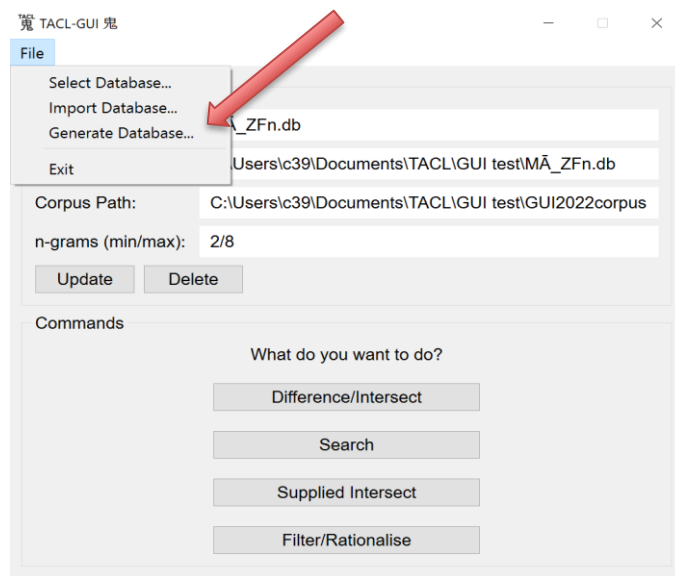
An Intersect test lists n-grams common to all sub-corpora in the test (as defined by the labels in the specified catalogue file). Intersect tests are particularly useful in finding debts of borrowing from an earlier text to a later text—for example, where a *sūtra* composed in China draws upon earlier genuine translations (as in the famous case of T156: <https://dazangthings.nz/cbc/text/1553/>); or when we wish to find which later authors (such as commentators) cite a text. We will explain how this process works by walking through a concrete example, which you can follow along as you read, using materials in your "starter kit".

We will take a known case as a simple example, which you can work through yourself (materials for this exercise are in your "starter kit"): the overlap between the *Qi chu san guan jing* 七處三觀經 T150A and the *Si yuan jing* 四願經 T735 (the portion of the text that Nattier calls "T735C"). This is our "EXAMPLE 2", as introduced above (we will later return to EXAMPLE 1, Zhu Fonian). For background information about this EXAMPLE 2, users can refer to Nattier (2008): 131-132, or <https://dazangthings.nz/cbc/text/274/>. Note that this new example will require us to use a different corpus and database.

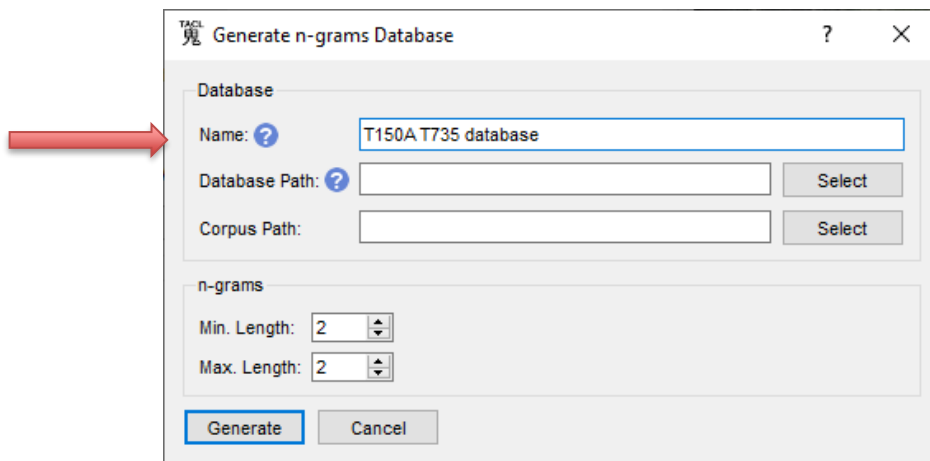
We imagine, then, that we are looking for possible sources of T150A. In this exercise, T735 stands in for a larger corpus of possible sources. (We use a single text on this side of the comparison only to make our example simple, and quick to run.)

We first need a corpus including T150 and T735 (your download packet includes such a corpus in the "corpora" subdirectory, named "T150A T735 corpus").

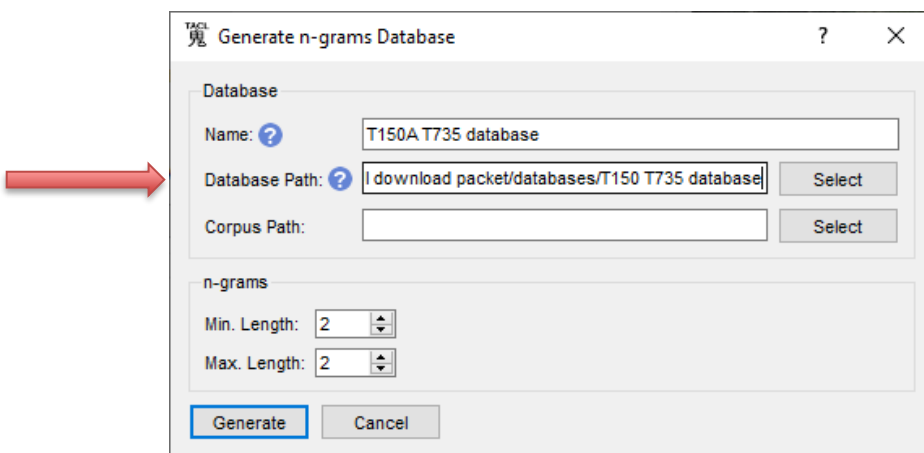
We then build a small, custom database for this corpus (however, you can skip this step if you have already downloaded and imported a full database). Once again, then, we select "Generate Database":



We then give the database a name:

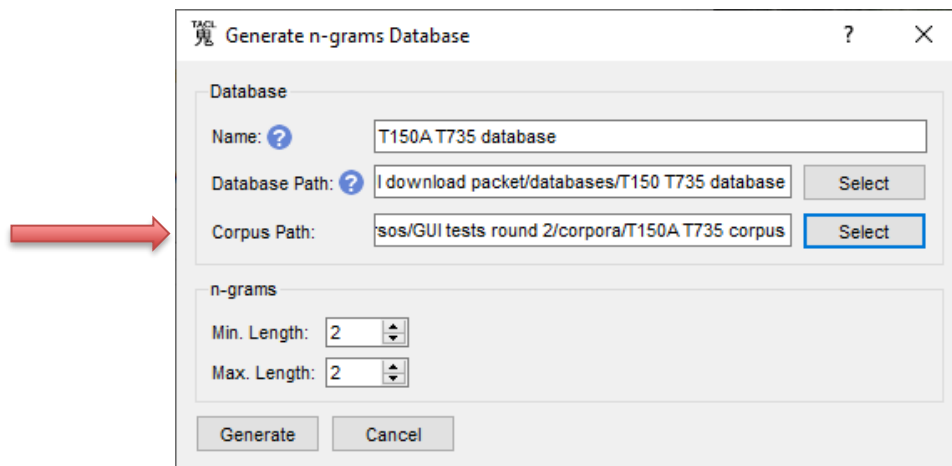


By navigating or typing, we input the path to the directory where we want to store the database, and type the database name again:

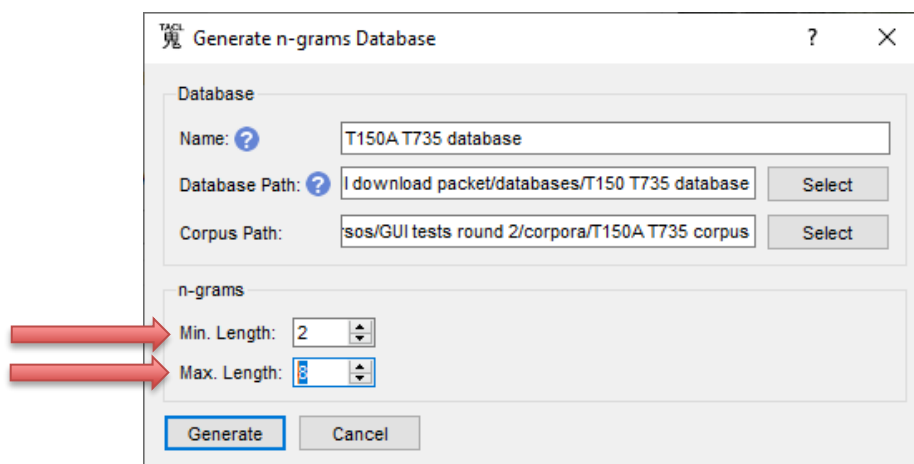




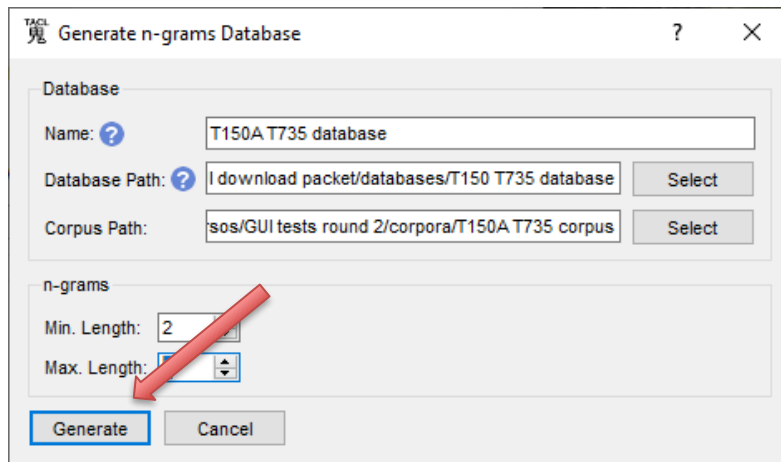
By navigating or typing, we tell the GUI the location of the corpus from which we want to generate the database:



Next, we set the minimum and maximum n-gram size, here, 2-grams to 8-grams:



Finally, we click "Generate", and wait for TACL to do its work.

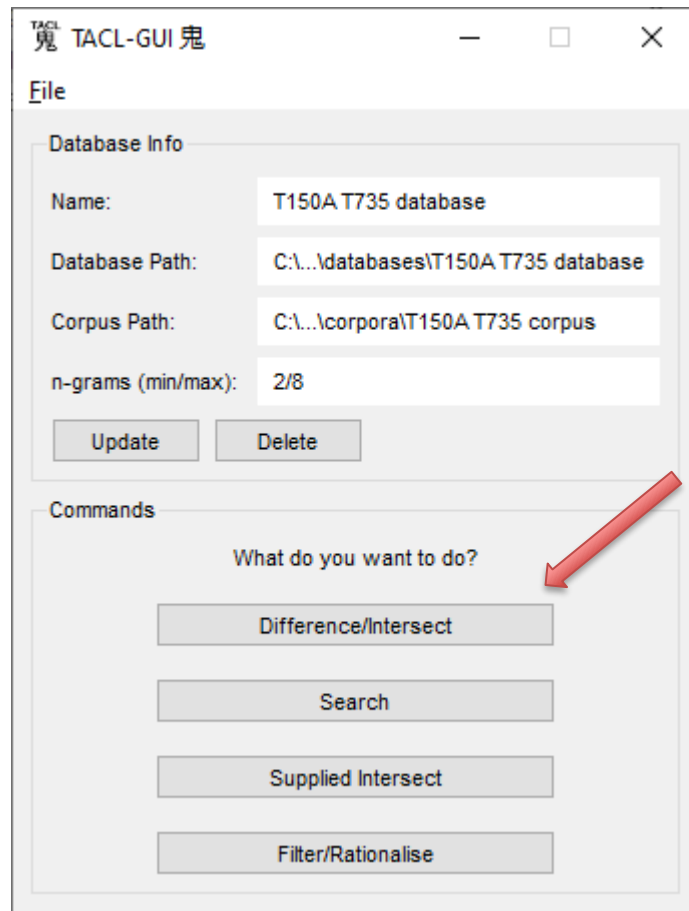


Next, we need a catalogue:

```
T0150A Qichu  
T0735 Siyuan
```

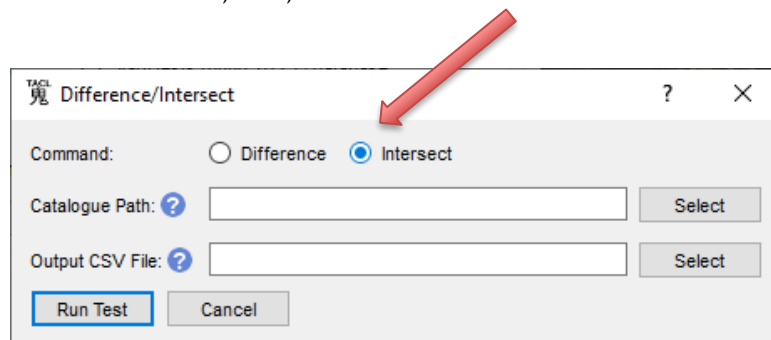
We save this file as e.g. "T150A vs T735 cat.txt" (in the "catalogues" subdirectory of your download packet).

We are now ready to run our actual Intersect test. At the main menu screen, we click "Difference/Intersect".

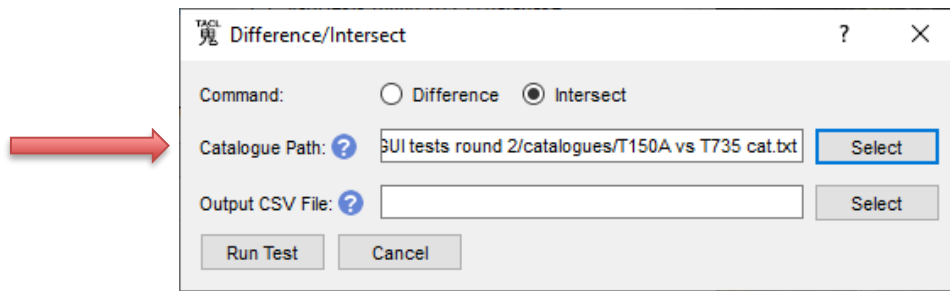


If you are following the steps laid out in this Manual, you should already have the correct database and corpus selected ("T150A T735 database", "T150A T735 corpus"), because we just constructed that database from that corpus. Later, however, when you are running other tests, you might come directly to this screen, and it is possible, if you have been working with various databases, that you might have the wrong database and corpus selected for the test you want to run. You should therefore always check at this point that you have the right database and corpus selected (as shown in the top portion of the screen in the screenshot above). If you do not, follow the instructions given above in the section "(3) Select a database" (p. 30).

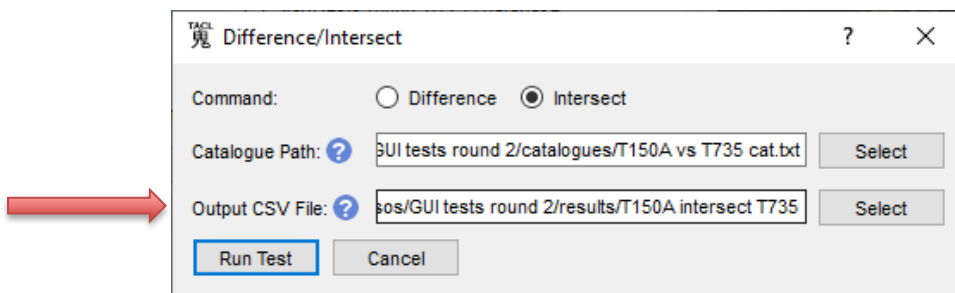
Next, in order to run the actual test, first, select "Intersect".



Supply the path to the catalogue either by navigating via "Select", or by typing the path.

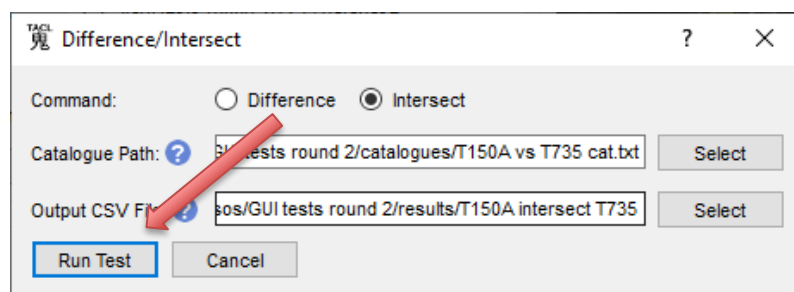


Once again, the items in the catalogue need to be assigned at least two labels. Items without labels will not be included in the operation.



Next, at "Output CSV File:", supply the path for the output .csv file in the same manner, by clicking "Select" and navigating to a directory, or by typing the path. Enter a filename for the results.

Then we click "Run Test".



While TACL is working, a window pops up with a "progress bar" (again, not a real one, that indicates actual progress, but just a "working" graphic). When the test is finished, the pop-up window will vanish, and you should be able to find the raw results .csv file in the location you supplied above.

Next, before you turn to human philological analysis of the raw results, we recommend that you always run raw results through a basic "Filter/Rationalize" operation. We explain this step in more

detail below (see "WORKFLOW STEP 6: Filter/Rationalize:", p. 63). For the moment, for the sake of this exercise, we will presume you have completed this step, and have a filtered results file called, e.g. "T150A intersect T735 filtered.csv".

Your TAFL results are now ready for human philological analysis. At this point, we recommend that you use a tool like Excel to view and analyse the results. We also explain this in more detail below (see "WORKFLOW STEP 7: Working with the results", p. 72 below). (Note! In Windows, if Excel is set as the default application for working with .csv files—as it usually is by default—you *cannot* open the results by simply clicking on them; they will come out garbled junk. Rather, follow the steps we describe below, p. 72.) Again, we will presume here that you have learnt this step; here, we aim just to show you what it looks like for the present exercise. You would import the results into Excel, then, and sort (using the sort protocol we recommend below for Intersect tests to find the sources of a text). You see something like this:

ngram	size	work	sigla	count	label	label count
生元謂識何等為生死靈識靈為生死靈識何等為生死靈識受行識為是八行識諦見至諦定為八如是為生死欲滅受行識	79	T0150A	元,宋,明	1	T150A	1
生死靈識何等為生死靈識靈為生死靈識何等為生死靈識受行識為是八行識諦見至諦定為八如是為生死欲滅受行識	79	T0735	CB,元,大,宋,宮,明,麗-CB	1	T735	1
生死靈識何等為生死靈識靈為生死靈識何等為生死靈識受行識為是八行識諦見至諦定為八如是為生死欲滅受行識	79	T0150A	大,大	0	T150A	1
識何等為思想要識所思想欲食能解欲食能斷欲食能自度如是為思想要識何等為生死識為六身生死識眼識生死識耳鼻	69	T0150A	元,宋,明	1	T150A	1
識何等為思想要識所思想欲食能解欲食能斷欲食能自度如是為思想要識何等為生死識為六身生死識眼識生死識耳鼻	69	T0735	CB,元,大,宋,宮,明,麗-CB	1	T735	1
識何等為思想要識所思想欲食能解欲食能斷欲食能自度如是為思想要識何等為生死識為六身生死識眼識生死識耳鼻	69	T0150A	大,大	0	T150A	1
畫為生死靈識何等為生死靈識受行識為是八行識諦見至諦定為八如是為生死欲滅受行識何等為生死味識所為生死	67	T0150A	明,明	1	T150A	1
畫為生死靈識何等為生死靈識受行識為是八行識諦見至諦定為八如是為生死欲滅受行識何等為生死味識所為生死	67	T0150A	明,明	0	T150A	1
生死要識所為生死欲食隨欲食能斷欲食能自度如是為生死要識何等為識身六衰識眼識耳鼻口身意識如是為識識何等	66	T0150A	元,宋,明	1	T150A	1
生死要識所為生死欲食隨欲食能斷欲食能自度如是為生死要識何等為識身六衰識眼識耳鼻口身意識如是為識識何等	66	T0735	CB,元,大,宋,宮,明,麗-CB	1	T735	1
生死要識所為生死欲食隨欲食能斷欲食能自度如是為生死要識何等為識身六衰識眼識耳鼻口身意識如是為識識何等	66	T0150A	大,大	0	T150A	1
何等為識靈受行識為八行識見至諦定為八如是為識靈欲受行如識識何等為識識知所識識因緣故生樂生喜意如是為味生	58	T0150A	CB,元,大,宋,明	1	T150A	1
何等為識靈受行識為八行識見至諦定為八如是為識靈欲受行如識識何等為識識知所識識因緣故生樂生喜意如是為味生	58	T0735	CB,元,大,宋,宮,明,麗-CB	1	T735	1
程度如是為要識如是比丘七處為覺知何等為七色習畫道味苦要是五陰各有七事何等為三觀識亦有七事得五陰成六	53	T0150A	CB,元,大,宋,明	1	T150A	1
程度如是為要識如是比丘七處為覺知何等為七色習畫道味苦要是五陰各有七事何等為三觀識亦有七事得五陰成六	53	T0735	CB,元,大,宋,宮,明,麗-CB	1	T735	1
識何等為思想要識所思想欲食能解欲食能斷欲食能自度如是為思想要識何等為生死識為六身生死識眼	44	T0150A	CB,大	1	T150A	1
識何等為思想要識所思想欲食能解欲食能斷欲食能自度如是為思想要識何等為生死識為六身生死識眼	44	T0150A	明,明,明	0	T150A	1
識諦見到諦定意為八如是為思想受行識何等為思想味識所為思想因緣生樂得意	35	T0150A	CB,大,明	1	T150A	1
識諦見到諦定意為八如是為思想受行識何等為思想味識所為思想因緣生樂得意	35	T0735	CB,元,大,宋,宮,明,麗-CB	1	T735	1
識諦見到諦定意為八如是為思想受行識何等為思想味識所為思想因緣生樂得意	35	T0150A	宋,宋	0	T150A	1

(We have also put a sorted set of results in Excel format in the "results" subdirectory, "T150A intersect T735 spreadsheet sorted.xlsx". Please have a look.)

In the first row here, we see that an n-gram, with size 79 (i.e. a 79-gram, a string 79 characters long) was found by TAFL as shared by both texts/groups under analysis (as defined by the catalogue). That is to say, this 79-gram occurs in both T150A and T735. And this is indeed the case—with the slight complication that in T150A, as we see in the "sigla" column, it only appears in the Song, Yuan and Ming witnesses, and not in the Taishō base text (大). It does not appear in the Taishō base text

because of a single variant reading (in red), which trips TACL up (remember that TACL finds only exact literal matches):

...生死習識。何等為生死盡識？**裁**[v.l. **裁, SYM**]盡為生死盡識。何等為生死欲盡受行識？為是八行識，諦見至諦定為八，如是為生死欲滅受行識。何等為生死味識？所為生死因緣生樂喜意，如是為生死味識。何等為生死..., T150A (II) 876b12-17.

...生死習識？何等為生死盡識？**裁**盡為生死盡識。何等為生死欲盡受行識？為是八行識，諦見至諦定為八，如是為生死欲滅受行識。何等為生死味識？所為生死因緣生樂喜意，如是為生死味識。何等為生死..., T735 (XVII) 537c2-6.

This long match is unique in the entire Taishō (run a CBETA search yourself to see, using a wildcard—the asterisk—for the v.l., i.e. you should search for 生死習識何等為生死盡識 \* 盡為生死盡識何等為生死欲盡受行識為是八行識諦見至諦定為八如是為生死欲滅受行識何等為生死味識所為生死因緣生樂喜意如是為生死味識何等為生死).

Obviously, such a long unique match cannot be a matter of coincidence, and this match is part of the evidence for the relation between the two texts already known from the scholarship cited above. If this were not already a known case, we would have just discovered a close intertextual relationship between T150A and T735.

Here is a second example for Intersect (EXAMPLE 3). Again, we work with a known case: the almost identical match between the \**Ekottarikāgama* 48.3 and the *Mile xiasheng jing* 彌勒下生經 T453 (see <https://dazangthings.nz/cbc/text/616/>). We imagine that we have a suspicion that *Ekottarikāgama* T125 might contain reuse of earlier text. Again, we use T453 as a proxy for a larger corpus, so that the test runs quickly for demonstration purposes. We describe this exercise more briefly, without screen shots and click-by-click instructions. We suggest that if you get stuck, you refer to the more detailed instructions for the T150A example above.

Thus, (1) we build a corpus including T125 and T453 (for this reason, T453 is included in "corpora/ZhuFonian corpus" in your download packet—we suggest you look in the corpus to check how it works, but we also suggest that you build your own, as an exercise).

(2) We generate or import a database for our corpus. We suggest you really do this for the "ZhuFonian corpus", since we will use the same corpus and database for further examples below (when we return to EXAMPLE 1).

(3) We construct a catalogue file defining T125 and T453 as two different corpora ("catalogues/Ekottarikagama vs T453 cat.txt"—we suggest you check the content of our catalogue as an example, but we also suggest you write your own, as an exercise).<sup>2</sup>

(4) We run an Intersect test on this corpus, database and catalogue. (5) We "Filter/Rationalize" the results. (6) We import into Excel, and sort the results (we have provided a sample as "results/Ekottarikagama intersect T453 spreadsheet sorted.xlsx", but we also suggest that you ultimately aim to do these steps of the exercise independently too).

If we have done all that correctly, the first result should be this 224-gram:

... 訓之所致也亦由四事因緣惠施仁愛利人等利爾時阿難彌勒如來當取迦葉僧伽梨著之是時迦葉身體奄然星散是時彌勒復取種種華香供養迦葉所以然者諸佛世尊有敬心於正法故彌勒亦由我所受正法化得成無上正真之道阿難當知彌勒佛第二會時有九十四億人皆是阿羅漢亦復是我遺教弟子行四事供養之所致也又彌勒第三之會九十二億人皆是阿羅漢亦復是我遺教弟子爾時比丘姓號皆名慈氏弟子如我今日諸聲聞皆稱釋迦弟子爾時彌勒與諸弟子說法汝等比丘當思惟無常之想樂有苦想計我無我想實有空想色變之想青瘀之想...

Once again, one of the results (this time in ĒA T125) is hidden from a search of the Taishō base text only by two v.l. (in red). But as we see in the "sigla" column of the "Filter/Rationalized" results, the same exact string is found in the Korean canon (CBETA siglum 麗). Again, this match is utterly unique in the entire canon (naturally, given that it is so long), and such a long match certainly cannot be coincidence. If we did not know it already from previous research, we would now be on our way to knowing that *Ekottarikāgama* 48.3 must have intertextual debts to T453 (or *vice versa*).

### Difference

A Difference test lists n-grams unique to each sub-corpus in a comparison, as defined by the labels in the specified catalogue file, or to only one side of the comparison (see "asymmetric Difference" below). Difference tests are particularly useful in finding recurring traits distinctive to one body of text against another, e.g. features of style (see further the TAFL Method Guide).

---

<sup>2</sup> Note: in our ZhuFonian corpus, the \**Ekottarikāgama* T125 appears as two texts, thus:

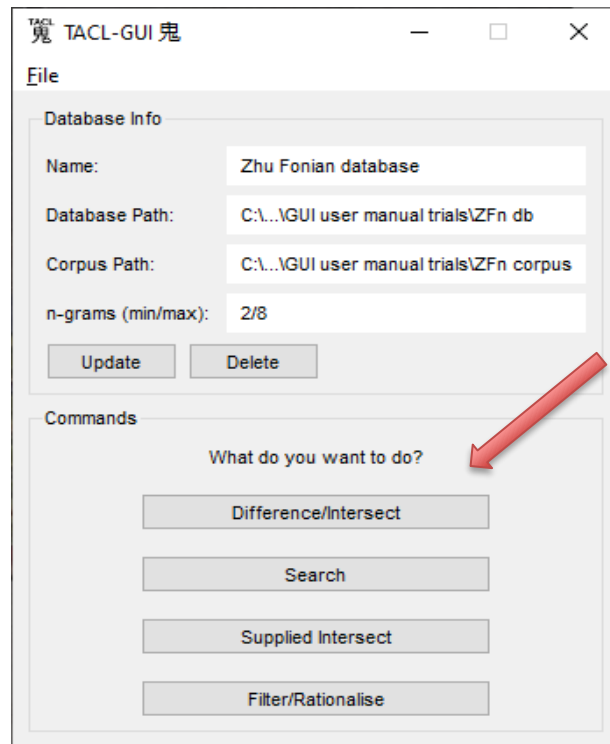
T0125-50-禮三寶品-4

T0125-minus-50.4

This is an example of the modifications to the Taishō texts typical of the "Radich" Taishō corpus for TAFL use. For the rationale, see <https://dazangthings.nz/cbc/text/4175/>.

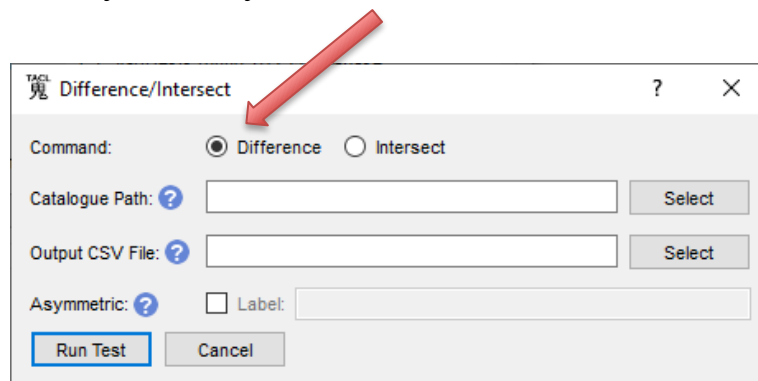
We will first walk you through the main steps in operating the GUI to run a Difference test, and then give an example as an exercise, which you can work through yourself.

To run a Difference test, click “Difference/Intersect” at the main menu screen.



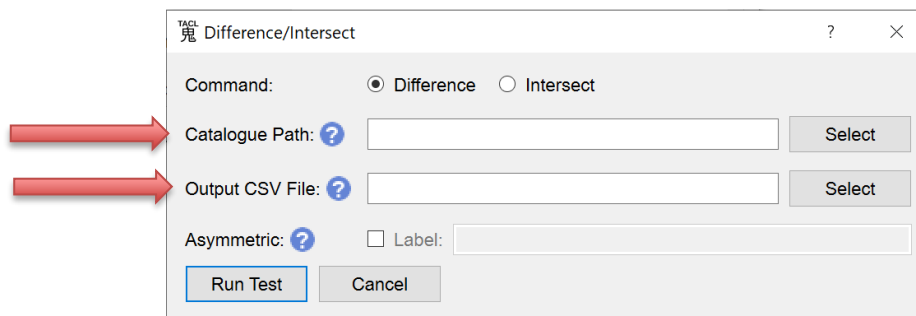
Once again, as we described for Intersect tests, it is important to check at this point that you have the correct database and corpus selected, as shown in the top portion of the screen. If you do not, follow the instructions given above in the section "(3) Select a database" (p. 30).

Select “Difference” (usually selected by default).





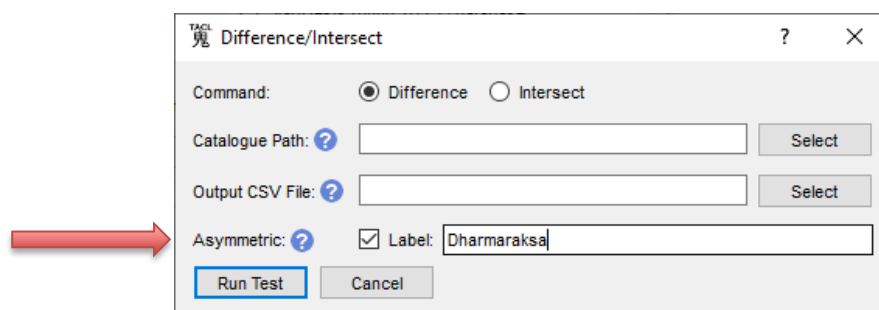
Aside from the database, the operation requires a catalogue (see above). As with “Intersect” (see above), supply the path to the catalogue either by navigating via “Select” or typing the path. Likewise, supply the path for the output .csv file, along with the intended filename.



### Asymmetric Difference

The "asymmetric" option in TACL Difference only finds and lists the n-grams unique to one of the sub-corpora. The advantage of this option is that we often do highly uneven comparisons, between one small corpus (e.g. a single text) and a much larger one (e.g. the entire canon). Restricting results to one side of the difference (usually the smaller) can vastly reduce the quantity of output, which can otherwise be overwhelming. Often, too, we are in fact only interested in one side of the comparison (e.g. what is distinctive about the texts of Zhi Qian, compared to all other translation texts?). This means that we need not generate vast quantities of results that we will not actually use.

To specify asymmetric Difference, check the “Asymmetric” box in the dialogue window and specify the label of the sub-corpus you want to focus on.



We begin, once more, with an artificially simple example—first, a symmetrical Difference (we will try the "asymmetrical" option later). We encourage you to work through this example (EXAMPLE 4) as you read.

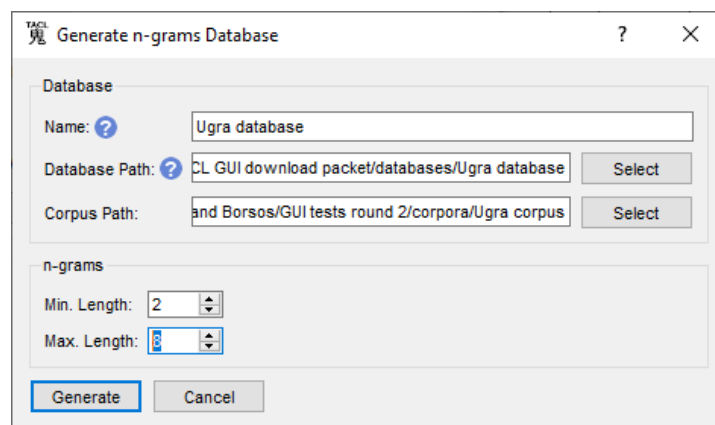
We imagine that we are trying to discover differences between the translation styles of (a) \*An Xuan and Yan Fotiao, against (b) Dharmarakṣa. Fortunately, both translation teams translated

versions of the same work: the *Ugra-pariṣcchā* (T322 by An Xuan and Yan Fotiao, T323 by Dharmarakṣa). Moreover, it is also clear (from unpublished TACL-based research) that Dharmarakṣa's group must have had the work of An Xuan and Yan Fotiao before them as they worked. Since the two texts largely overlap in content, differences between them are more likely to betray the styles distinctive of each team against the other; because Dharmarakṣa knew the earlier translation, moreover, in the case of Dharmarakṣa, the differences are even likely to be deliberate or conscious.

Once more, we have provided the corpus, catalogue, and sample results files in your download packet, but we recommend that you try to independently achieve all steps to run the test yourself. In this example, we will simplify the screenshots somewhat, on the assumption that the GUI is already becoming familiar to you.

We build a corpus, containing T322 and T323 ("corpora/Ugra corpus" in your starter kit).

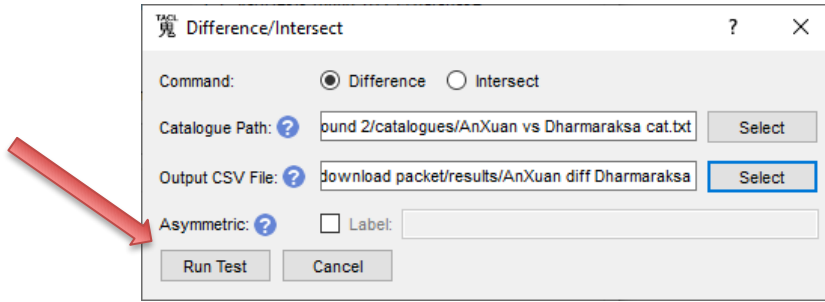
Unless we have a full database including those texts already, we build a custom database for that corpus.



We make a catalogue file defining the two corpora against one another ("catalogues/AnXuan vs Dharmaraksa cat.txt"):

```
T0322 AnXuan
T0323 Dharmaraksa
```

We go to the Difference/Intersect screen, tell the GUI where to find the catalogue and where to save the results file, and run the test.

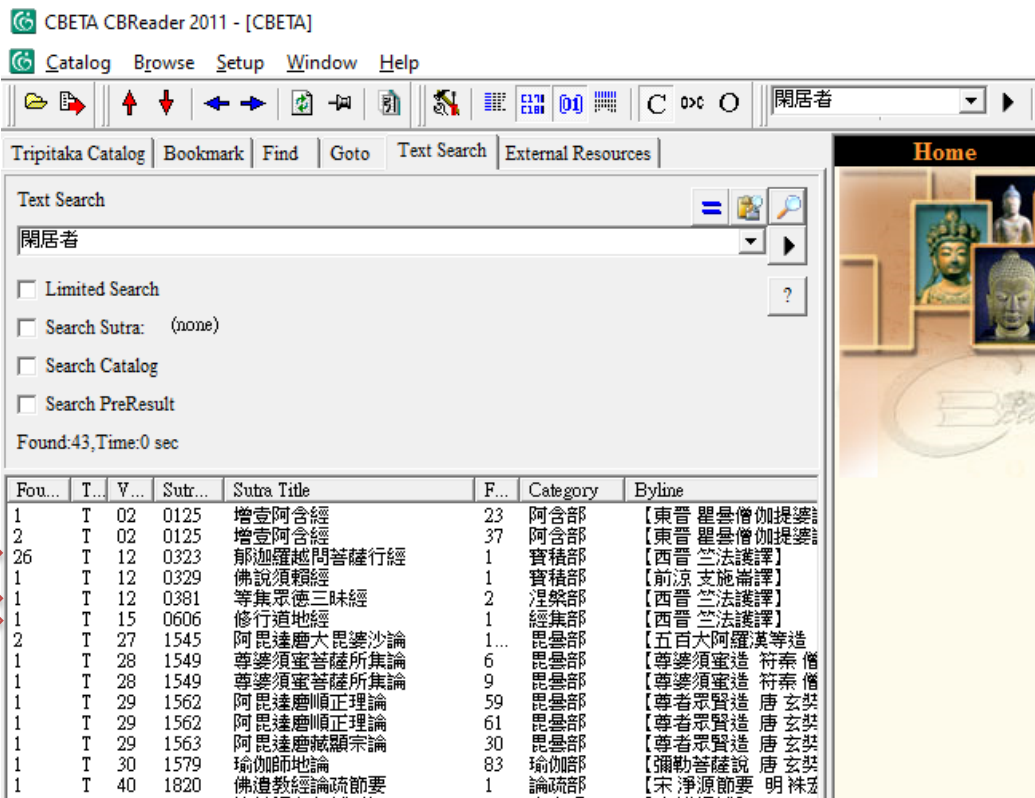


In this case, we do not select the option for asymmetrical Difference. We then run the results through a basic "Filter/Rationalize" operation, import the results into Excel, and sort them (using the sort protocol recommended below for tests for style: first by count, in descending order; then by size, in ascending order: see "results/AnXuan diff Dharmaraksa spreadsheet.xlsx", worksheet = tab entitled "symmetrical").

If we have done all that correctly, one of the first results we see in our sorted Excel results file (Row 28) is 閑居者, which occurs 26 times in Dharmarakṣa's T323 (and by definition, because this is a Difference test, never occurs in T322):

1	ngram	size	work	sigla	count	label	label count
17	以不	2	T0322	元, 宋, 宮	41	AnXuan	42
18	在閑	2	T0323	CB, 元, 大, 宋, 宮, 明, 禪-CB, 麗-CB	41	Dharmaraksa	41
19	以為	2	T0322	元, 宋, 宮	35	AnXuan	35
20	以為	2	T0322	CB, 大, 明, 禪-CB, 麗-CB	34	AnXuan	35
21	一者	2	T0323	宋	33	Dharmaraksa	33
22	一者	2	T0323	CB, 元, 大, 宮, 明, 禪-CB, 麗-CB	32	Dharmaraksa	33
23	三者	2	T0323	CB, 元, 大, 宋, 宮, 明, 禪-CB, 麗-CB	32	Dharmaraksa	32
24	具足	2	T0323	CB, 元, 大, 宋, 宮, 明, 禪-CB, 麗-CB	32	Dharmaraksa	32
25	若此	2	T0322	CB, 元, 大, 宋, 宮, 明, 禪-CB, 麗-CB	31	AnXuan	31
26	家開	2	T0322	CB, 大, 明, 禪-CB, 麗-CB	26	AnXuan	26
27	家開士	3	T0322	CB, 大, 明, 禪-CB, 麗-CB	26	AnXuan	26
28	閑居者	3	T0323	CB, 元, 大, 宋, 宮, 明, 禪-CB, 麗-CB	26	Dharmaraksa	26
29	家開	2	T0322	元, 宋, 宮	25	AnXuan	26
30	家開士	3	T0322	元, 宋, 宮	25	AnXuan	26
31	出家菩	3	T0323	CB, 元, 大, 宋, 宮, 明, 禪-CB, 麗-CB	25	Dharmaraksa	25
32	出家菩薩	4	T0323	CB, 元, 大, 宋, 宮, 明, 禪-CB, 麗-CB	25	Dharmaraksa	25
33	施與	2	T0323	CB, 元, 大, 宋, 宮, 明, 禪-CB, 麗-CB	24	Dharmaraksa	24
34	眾生	2	T0322	元, 宋, 宮, 明	22	AnXuan	22
35	復次	2	T0323	CB, 元, 大, 宋, 宮, 明, 禪-CB, 麗-CB	22	Dharmaraksa	22
36	次長	2	T0323	CB, 元, 大, 宋, 宮, 明, 禪-CB, 麗-CB	22	Dharmaraksa	22

And indeed, this 3-gram also appears elsewhere in Dharmarakṣa (and a few other places), though very seldom.



In other words, though this result is very slight compared to many things TACL has the power to discover, even with this artificially simple example we are indeed on our way to discovering real stylistic differences distinguishing Dharmarakṣa against his predecessors.

We now turn to a second example for a Difference test, and start addressing something closer to a real research problem. For this purpose, we return to EXAMPLE 1, Zhu Fonian and the \*Ekottarikāgama.

A key use of TACL “Difference” is to find possible stylistic markers distinctive of a translator (or translation group, or some other “author”), against some meaningful point of comparison. We can illustrate this through tests exploring the long-standing problem of whether the extant \*Ekottarikāgama T125 is more likely to have been translated by (a group centring upon) Zhu Fonian, or Saṅghadeva (see [here](#) for notes on some of the relevant scholarship). This time, however, we will concatenate the Difference results with further tests, to yield greater analytical power.

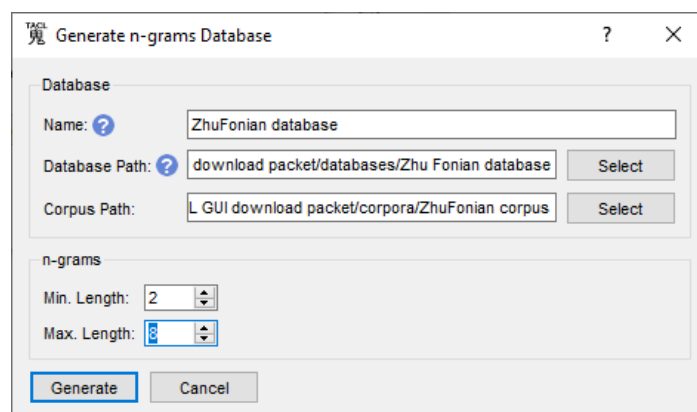
We begin with a test of the Difference between Saṅghadeva and Zhu Fonian. We use a TACL catalogue file that identifies the reliable corpus for both translators. For the moment, we bracket Ekottarikāgama T125 itself out of the "Zhu Fonian" corpus, since it is the uncertain text that is the object of the test. We achieve this by leaving T125 in the catalogue, but giving it no label (so that TACL ignores it).

T0001-30-世記經 ZhuFonian  
 T0001-minus-30 ZhuFonian  
 T0125-50-禮三寶品-4  
 T0125-minus-50.4  
 T0194 ZhuFonian  
 T0212 ZhuFonian  
 T0309 ZhuFonian  
 T0384 ZhuFonian  
 T0385 ZhuFonian  
 T0656 ZhuFonian  
 T1428 ZhuFonian  
 T1464 ZhuFonian  
 T1505 ZhuFonian  
 T1543 ZhuFonian  
 T1549 ZhuFonian  
 T2045 ZhuFonian

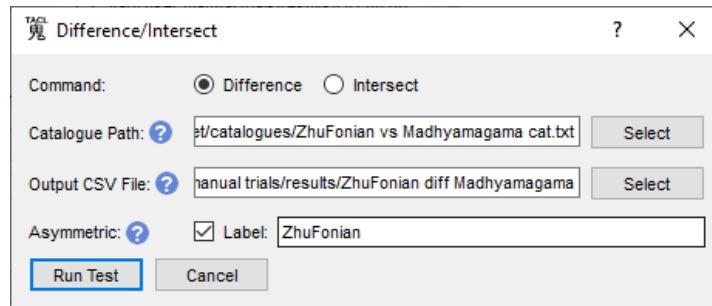
T0026-whole Saṅghadeva

For the sake of simplicity, we will imagine that we are at first only interested in the Zhu Fonian side of this comparison. (In reality, a fuller investigation of the problem, if we did not already know the answer, would require systematic testing of both sides of the comparison; see also Radich 2017a, Radich and Anālayo 2017.) We therefore run an asymmetric Difference for the "ZhuFonian corpus", against Saṅghadeva. This test helps us discover recurring n-grams particular to Zhu Fonian, which are never found in Saṅghadeva's *Madhyamāgama*. Among these n-grams, we are very likely to find strong stylistic markers distinguishing Zhu Fonian from Saṅghadeva.

For this test, we must first have a database for the "ZhuFonian corpus" (see the "ZhuFonian corpus" in the "corpora" subdirectory of your download packet), or for a larger corpus that includes all the works in that corpus (such as the "Radich Taisho corpus"). If you did not generate the ZhuFonian database when it was mentioned above, we suggest you generate it now.

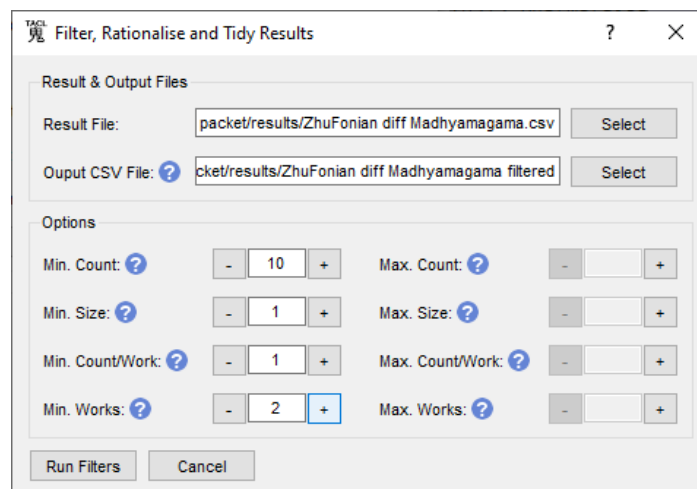


We then run a Difference—this time, an asymmetric Difference—between ZhuFonian and the *Madhyamāgama*/Saṅghadeva. (Be warned: This test takes somewhat longer than the other tests we have done so far.)



Below, we will "concatenate" the results of this Difference operation with other TACL operations (see "WORKFLOW STEP 5: Supplied Intersect", p. 55). For such a concatenated test, we need to work with the raw results we have just generated (i.e. not run the results through Filter/Rationalize).

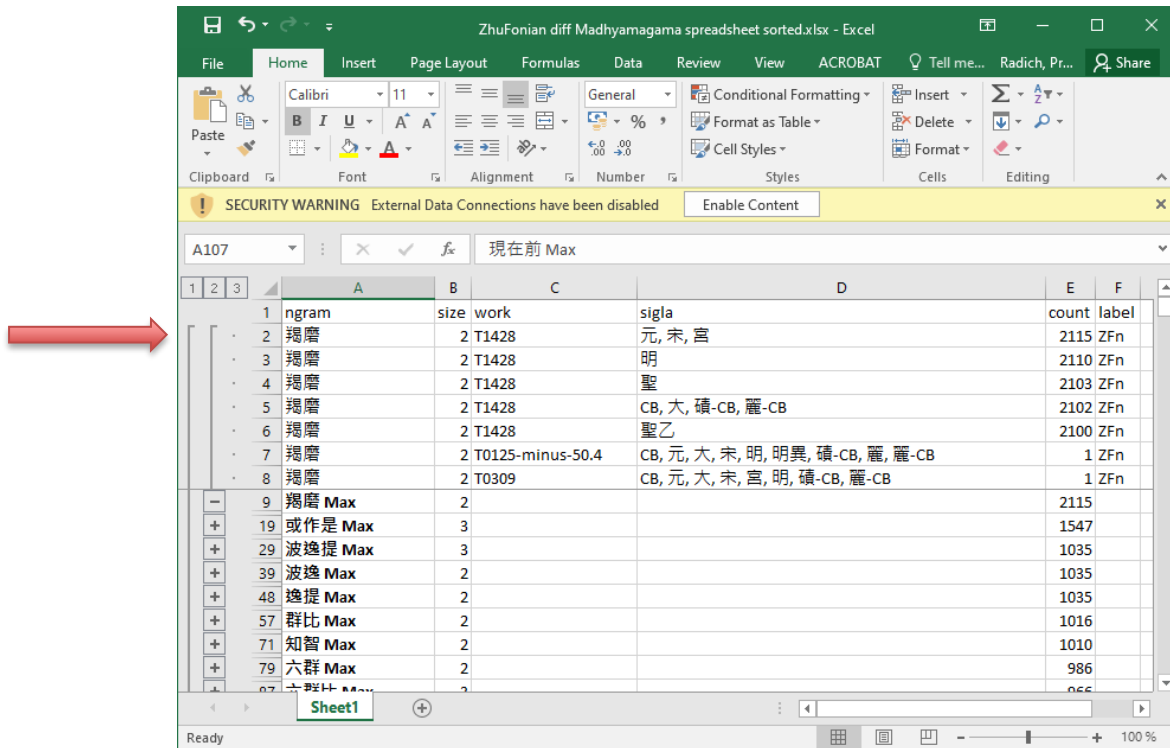
Meanwhile, however, let us assume for the sake of our exercise that we are already directly interested in the results of this test alone. As usual, then, before we proceed to human philological analysis, we process the results with a basic "Filter/Rationalize" operation. This time, however, the results will be too copious for Excel (or a human) to process in their entirety, so we also specify a minimum count of 10, and a minimum of 2 works (see more below under "WORKFLOW STEP 6: Filter/Rationalize:", p. 63), in order to remove items only occurring relatively rarely in the ZhuFonian corpus.



We then import into Excel, and sort (this time using the sort protocol recommended for tests for style: first by count, in descending order, then by size, in ascending order; see "WORKFLOW STEP 7:

Working with the results", p. 72). Again, we have provided an example of such a sorted spreadsheet, but we recommend that you ultimately also try to create your own.

If we have done that correctly, we will see results that begin like this ("ZhuFonian diff Madhyamagama spreadsheet sorted.xlsx"):



And indeed, the first item, 羯磨, is found in three Zhu Fonian works—very abundantly in T1428, but once each only in T125 and T309—and never in *Madhyamāgama*. (This can be seen from the above screenshot, where we have expanded the results for 羯磨 by clicking on the "plus" sign next to the n-gram; you can also confirm it for yourself by running a search in CBETA.) The next item, 或作是~ (which we see in context is variously 或作是說, 或作是語, 或作是論, etc.) is found in six Zhu Fonian works (the overwhelming majority in T1549), but never in *Madhyamāgama*. We are already on our way to discovering possible stylistic differences between Zhu Fonian and Saṅghadeva.

#### WORKFLOW STEP 5: Supplied Intersect

Supplied Intersect lists n-grams common to all results files from prior rounds of TACL tests (prior Difference or Intersect tests).

In other words, Supplied Intersect always follows at least two previous TACL operations. Note that all the results files must be raw, i.e. must not themselves have been submitted to any Filter/Rationalize operations.

Our example here continues from the second example for Difference above, i.e. EXAMPLE 1, Zhu Fonian and the \**Ekottarikāgama*.

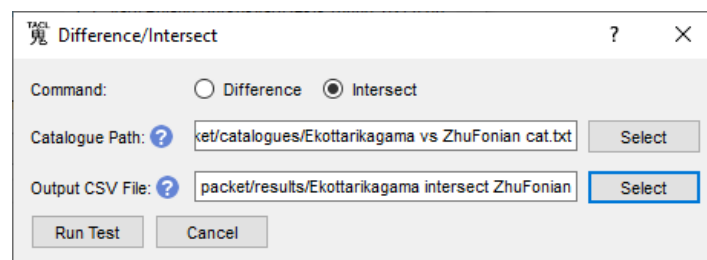
First, the asymmetric Difference operation between the ZhuFonian corpus and *Madhyamāgama* is but one of the two preparatory operations for the investigation whether *Ekottarikāgama* might have been by ZhuFonian or Saṅghadeva. The Difference operation gives us n-grams that are only found in ZhuFonian works but not *Madhyamāgama*.

We also need another test to give us another side of the picture: another test that intersects *Ekottarikāgama* with the ZhuFonian-minus-*Ekottarikāgama* corpus, yielding n-grams that are found in both *Ekottarikāgama* and ZhuFonian. That test operates out of the same "ZhuFonian corpus" and database as the test above, but uses a different catalogue ("catalogues/*Ekottarikagama* vs ZhuFonian cat.txt"):

```
T0125-50-禮三寶品-4 Ekottarikagama
T0125-minus-50.4 Ekottarikagama

T0001-30-世記經 ZhuFonian-other
T0001-minus-30 ZhuFonian-other
T0194 ZhuFonian-other
T0212 ZhuFonian-other
T0309 ZhuFonian-other
T0384 ZhuFonian-other
T0385 ZhuFonian-other
T0656 ZhuFonian-other
T1428 ZhuFonian-other
T1464 ZhuFonian-other
T1505 ZhuFonian-other
T1543 ZhuFonian-other
T1549 ZhuFonian-other
T2045 ZhuFonian-other
```

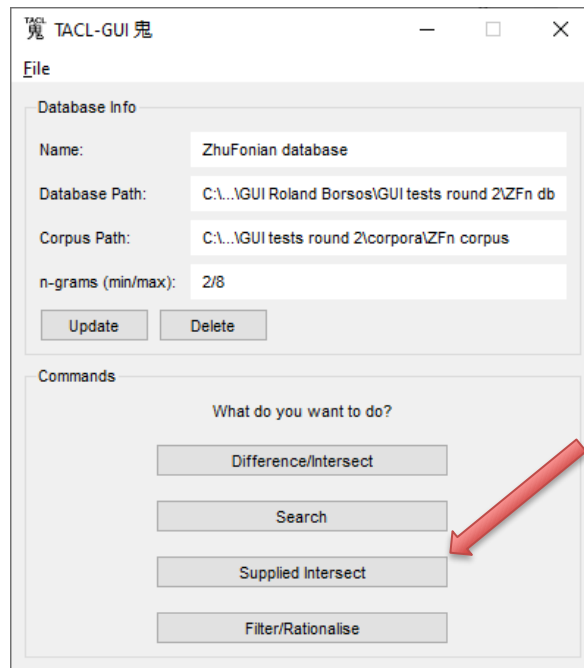
With these materials, we run our Intersect test:



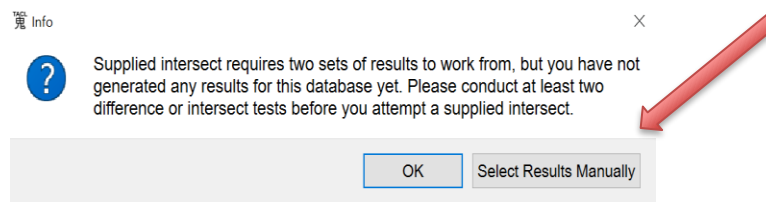


Finally, we find the "Supplied Intersect" of both of the raw results files from the previous two tests, using "Supplied Intersect". This step, combining results of the previous two operations, finds n-grams that are *both* (a) found in both *Ekottarikāgama* and the ZhuFonian corpus; but (b) *not* found in *Madhyamāgama*.

To start, go to the main menu screen, and click "Supplied Intersect".



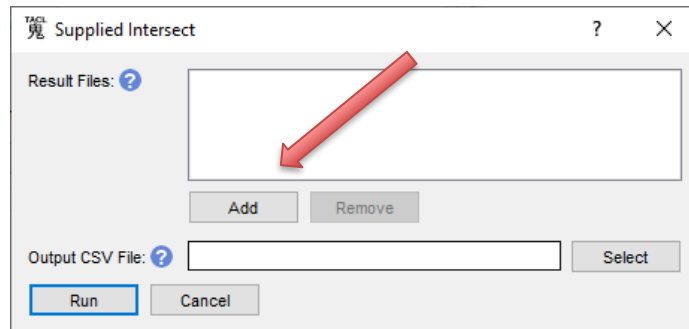
After you click Supplied Intersect, if you have not previously generated at least two results files from the database you selected, you might see the following message:



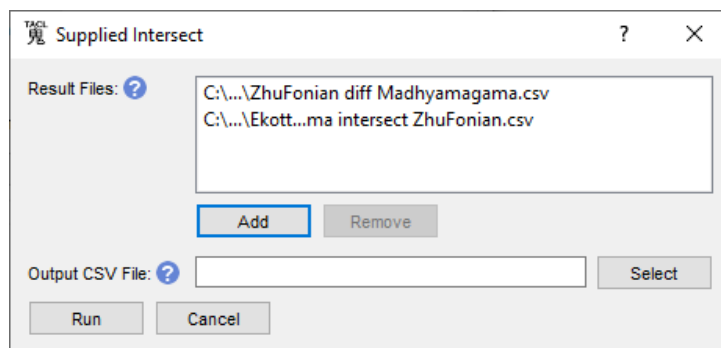
However, if you have at least two sets of results from other databases, you can still proceed. Simply click on "Select Results Manually".

Alternatively, if you indeed have not yet prepared any results files beforehand, click "OK". The GUI will lead you directly to the Difference/Intersect window, and you can then first conduct the Difference/Intersect tests needed to produce the raw results to supply to your Supplied Intersect operation.

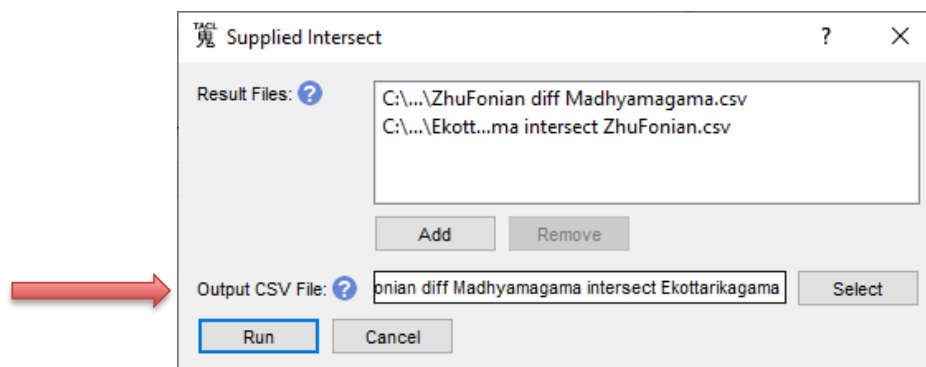
Assuming you have prepared raw results files by either of these methods, proceed to the window for Supplied Intersect. Click "Add" to select the results files you intend to intersect.



Then navigate to the directory and click on the raw output .csv files or type in the path and filenames:



Then click "Select" in the "Output CSV File" field, and enter a path and filename for the output results file (by navigating or typing):



Finally, click "Run".

To summarize the sequence of operations behind this example for Supplied Intersect, we took the following steps:

1. asymmetric ZhuFonian diff *Madhyamāgama*
2. *Ekottarikāgama* intersect ZhuFonian
3. "supplied" intersect (1) and (2)

As with the results of other tests, raw results from a Supplied Intersect should finally be processed with a basic Filter/Rationalize operation, before they are imported to Excel for human philological analysis (see again below, "WORKFLOW STEP 6: Filter/Rationalize:" p. 63, and "WORKFLOW STEP 7: Working with the results" p. 72).

This example (EXAMPLE 1) is not hypothetical. Difference tests between *Ekottarikāgama* and *Madhyamāgama* alone are at the base of the work presented in Radich and Anālayo (2017), which shows that *Ekottarikāgama* and *Madhyamāgama* systematically use different ways of rendering the same recurring elements in their Indic *Vorlagen*, and therefore, that it is overwhelmingly unlikely that *Ekottarikāgama* is by the same translator as *Madhyamāgama*, namely, Saṅghadeva (2017). Radich then followed up with a second publication (2017a) in which he worked with results similar to those yielded by the Supplied Intersect we have just outlined, and showed that many recurring stylistic features of *Ekottarikāgama* are shared with other core Zhu Fonian texts, and never found in *Madhyamāgama*/Saṅghadeva. In combination, and viewed in the light of external evidence as well, these findings mean that it is almost certain that our extant *Ekottarikāgama* T125 is by Zhu Fonian (and his collaborators), and not by Saṅghadeva.

### Search

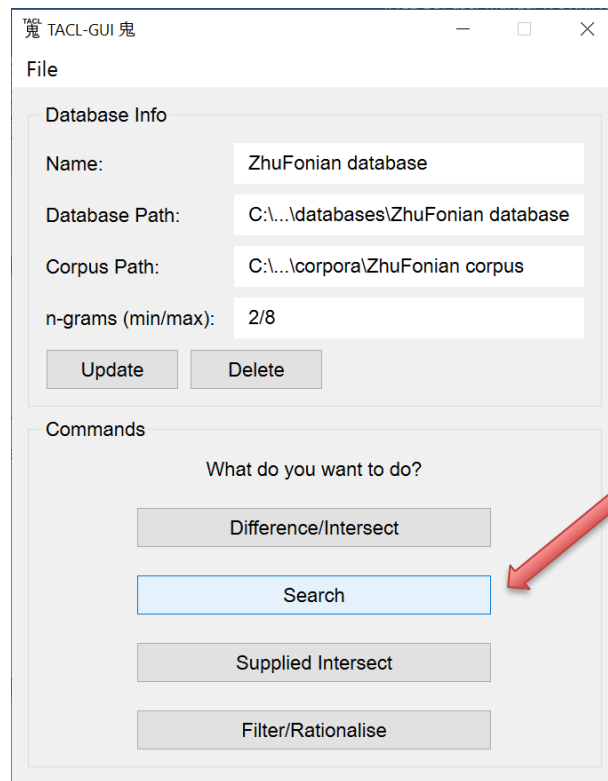
In TACL Search, the user supplies a list of n-grams, and Search outputs all instances of those n-grams that occur within each of a set of labelled works, as stipulated in a catalogue file.

The Search function requires the user to input an "n-grams File". You should therefore first prepare this file, which list the n-grams you want to search for. Format the file with one n-gram per line, and save it in text-only (.txt) format. As our example, we will use some of the n-grams that distinguish the Zhu Fonian style (including *Ekottarikāgama*) from the Saṅghadeva style represented by *Madhyamāgama*. A sample n-grams file (artificially short) might look like this (see "ZhuFonian 10 ngrams.txt" in the "starter kit"):

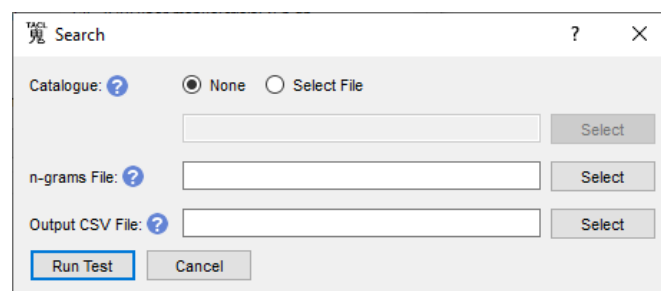
一時佛在  
在一面坐  
村落

閻浮提  
釋迦文  
苦出要諦  
四意止  
阿闍世  
典兵寶  
在閑靜

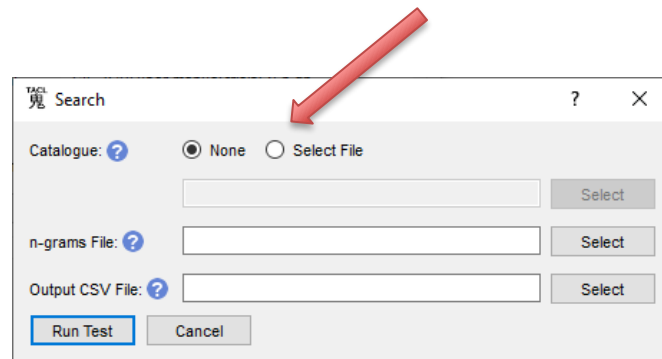
As long as you have a corpus and corresponding database, you are now ready to run your TACL Search. At the TACL GUI main menu, click "Search".



You will then see this screen:

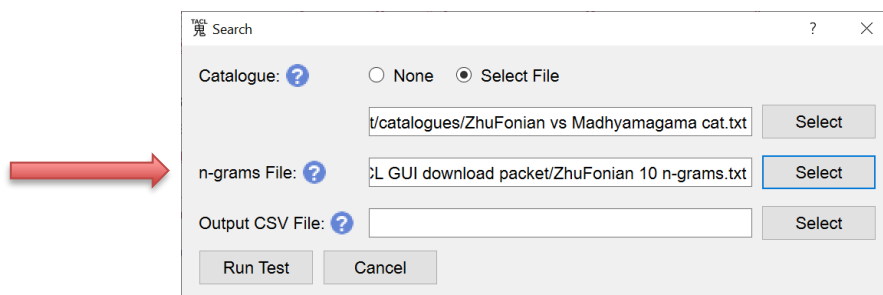


The catalogue file is optional. This means that the user can supply a catalogue, which sorts words of interest into various labels (groups). The Search will then accordingly sort the n-grams it finds by these labels. If you wish to use a catalogue file, click "Select File".

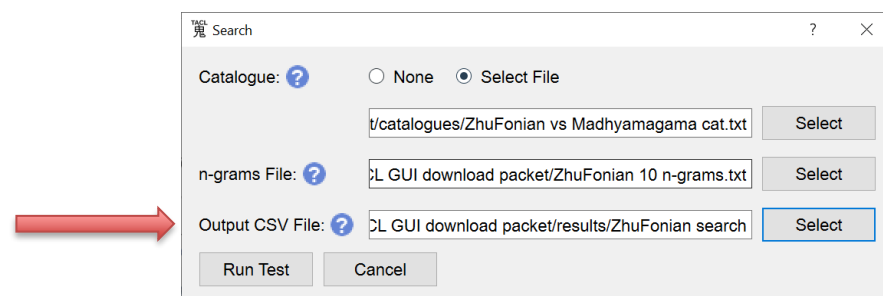


As above, give the path to the catalogue either by navigating to the directory via "Select", or typing the path directly. For our example, we will use "ZhuFonian vs Madhyamagama cat.txt" (in the "catalogues" directory of your "starter kit"), which distinguishes two groups: (a) the whole Zhu Fonian corpus, including T125; and (b) the *Madhyamāgama*.

As already mentioned, Search requires a list of n-grams. The user therefore must select an n-grams file. This means that in the box for "n-grams File", point the GUI to the n-grams file that we mentioned above, "ZhuFonian 10 n-grams.txt".



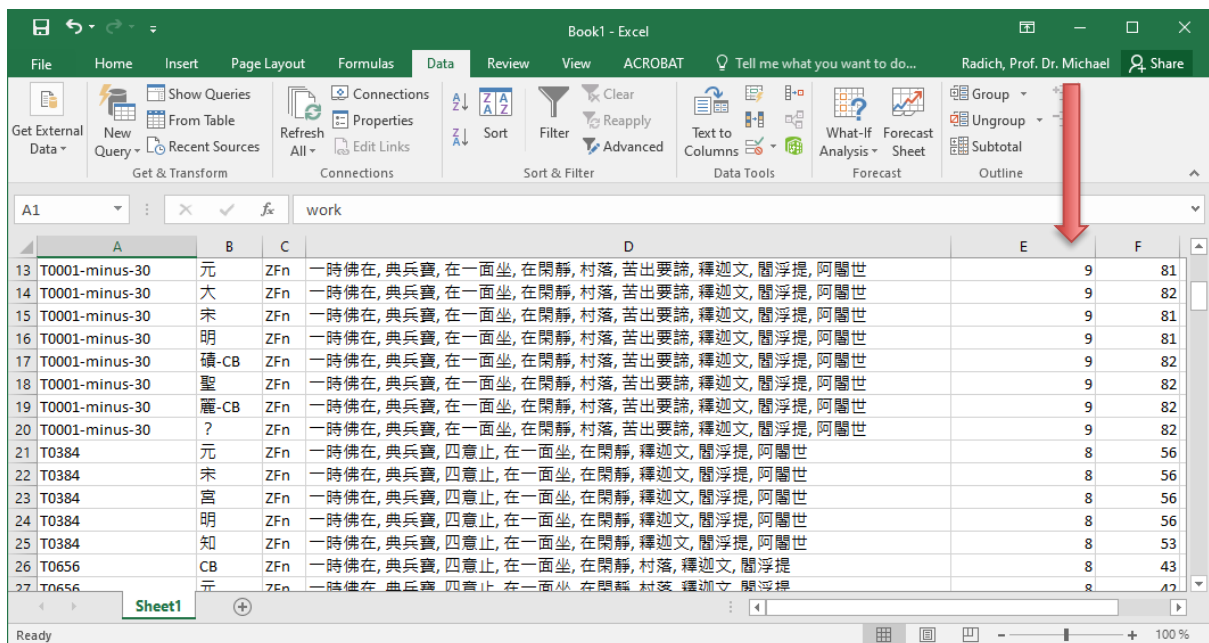
Finally, in the "Output CSV File" field, input a path and filename for the output CSV file using by navigating and/or typing, as above.



Then click "Run Test".

Note that we do not run Search results through "Filter/Rationalize". The files generated by Search differ in format from other results files, and it is simply impossible to run them through Filter/Rationalize. If you attempt to do so, you will get an error. Another way of thinking of this point is that default processes in the background have already tidied Search results, in a manner similar to what is achieved by Filter/Rationalize for those other results.

If we have done all that correctly, we then import into Excel, and sort by the protocol recommended below for Search results (on number of n-grams, in descending order). We then see something like this:



	A	B	C	D	E	F
13	T0001-minus-30	元	ZFn	一時佛在, 典兵寶, 在一面坐, 在閑靜, 村落, 苦出要諦, 釋迦文, 闍浮提, 阿闍世	9	81
14	T0001-minus-30	大	ZFn	一時佛在, 典兵寶, 在一面坐, 在閑靜, 村落, 苦出要諦, 釋迦文, 闍浮提, 阿闍世	9	82
15	T0001-minus-30	宋	ZFn	一時佛在, 典兵寶, 在一面坐, 在閑靜, 村落, 苦出要諦, 釋迦文, 闍浮提, 阿闍世	9	81
16	T0001-minus-30	明	ZFn	一時佛在, 典兵寶, 在一面坐, 在閑靜, 村落, 苦出要諦, 釋迦文, 闍浮提, 阿闍世	9	81
17	T0001-minus-30	禪-CB	ZFn	一時佛在, 典兵寶, 在一面坐, 在閑靜, 村落, 苦出要諦, 釋迦文, 闍浮提, 阿闍世	9	82
18	T0001-minus-30	聖	ZFn	一時佛在, 典兵寶, 在一面坐, 在閑靜, 村落, 苦出要諦, 釋迦文, 闍浮提, 阿闍世	9	82
19	T0001-minus-30	瞿-CB	ZFn	一時佛在, 典兵寶, 在一面坐, 在閑靜, 村落, 苦出要諦, 釋迦文, 闍浮提, 阿闍世	9	82
20	T0001-minus-30	?	ZFn	一時佛在, 典兵寶, 在一面坐, 在閑靜, 村落, 苦出要諦, 釋迦文, 闍浮提, 阿闍世	9	82
21	T0384	元	ZFn	一時佛在, 典兵寶, 四意止, 在一面坐, 在閑靜, 釋迦文, 闍浮提, 阿闍世	8	56
22	T0384	宋	ZFn	一時佛在, 典兵寶, 四意止, 在一面坐, 在閑靜, 釋迦文, 闍浮提, 阿闍世	8	56
23	T0384	宮	ZFn	一時佛在, 典兵寶, 四意止, 在一面坐, 在閑靜, 釋迦文, 闍浮提, 阿闍世	8	56
24	T0384	明	ZFn	一時佛在, 典兵寶, 四意止, 在一面坐, 在閑靜, 釋迦文, 闍浮提, 阿闍世	8	56
25	T0384	知	ZFn	一時佛在, 典兵寶, 四意止, 在一面坐, 在閑靜, 釋迦文, 闍浮提, 阿闍世	8	53
26	T0656	CB	ZFn	一時佛在, 典兵寶, 四意止, 在一面坐, 在閑靜, 村落, 釋迦文, 闍浮提	8	43
27	T0656	元	ZFn	一時佛在, 典兵寶, 四意止, 在一面坐, 在閑靜, 村落, 釋迦文, 闍浮提	8	42

This shows that nine of the n-grams we are looking for are found in the *Dirghāgama* T1; eight are found in T384; and so on. By contrast, none of the n-grams on our list are found in *Madhyamāgama*, and they therefore do not appear in the Search results. Readers who are interested in seeing how evidence like this can be used in making an argument about ascription might compare these results to the table in Radich and Anālayo (2017): 228-229.

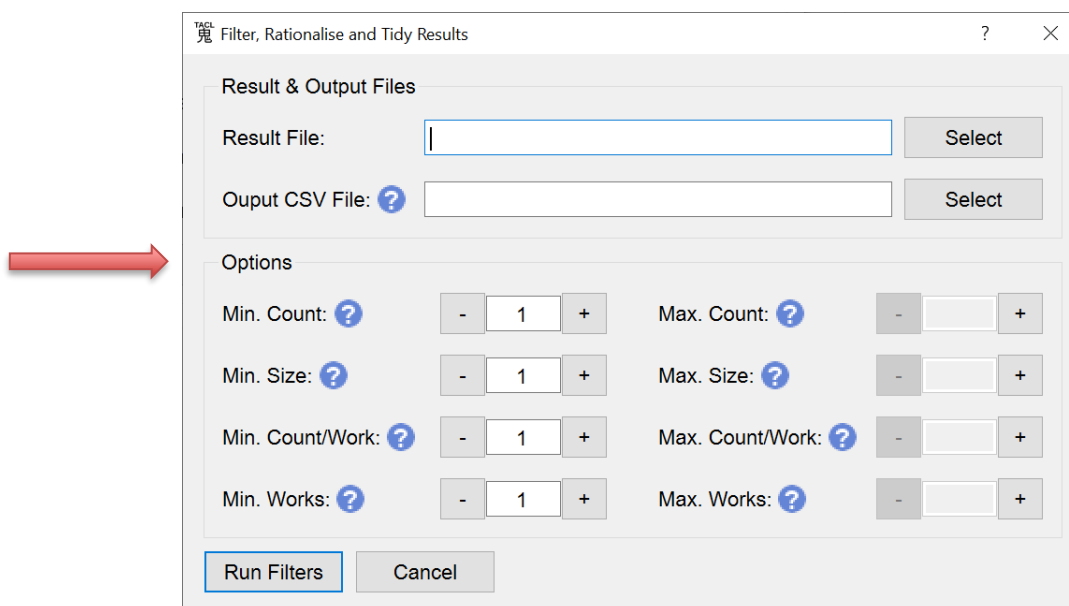
As we describe further in the TACL Methods Guide, Search results can be particularly powerful when we have a list of n-grams characteristic of the style of a given translator or group, and want to conduct an open search, as far afield as possible, to find where those markers might be

concentrated *outside* the known corpus of that translator/group—that is, to find texts that might be hitherto unknown works by known persons.

#### WORKFLOW STEP 6: Filter/Rationalize:

As we have already mentioned several times, with all results except Search results, we recommend that users pass raw results through "Filter/Rationalize", as the last step before beginning human philological analysis. This step must be left until last, because Supplied Intersect tests performed on results filtered for counts, etc., can produce false results.

Users can think of the Filter/Rationalize functions as falling into two sets. First, there is a set of “tidying” operations that are hardwired into the GUI, and are achieved in the background without the user doing anything. Second, there is also a set of parameters that users can see and adjust via the Filter/Rationalize screen.



We will first describe the background “tidying” operations, which are hardwired into the GUI. Because Difference and Intersect texts differ in purpose, the background "tidying" operations work differently for each. Intersect results are subject to four types of tidying operations, while Difference results undergo only two of these four operations. In order to show all four operations, we will therefore use as our example an Intersect test: the unfiltered results and filtered results in “T150A intersect T735 spreadsheet.xlsx” (in the "results" directory of the "starter kit"; once again, this was our "EXAMPLE 2"). To show the difference made by these various operations, we present on different tabs of the spreadsheet results that have or have not been through the operations of "Filter/Rationalize": respectively, unfiltered results without any sorting applied; unfiltered results, sorted by size; unfiltered results, sorted by n-gram; and then the filtered results. Below, we will walk you through the comparisons this allows.

For both Difference and Intersect tests alike, Filter/Rationalize groups on a single row of the table all witnesses that feature the same n-gram with the same count. Raw results otherwise have a single row per witness, which would mean that the human user would see evidence of "the same" thing—the number of hits for an n-gram in a given work—a separate time for each witness, and make results tables several times longer. In the “T150A intersect T735 spreadsheet.xlsx”, we can see the effect of this tidying operation by comparing results "unfiltered, sorted by n-gram" (on the tab with that name) with the filtered results (on the tab “default filters, sorted by size”) (see screenshots below). Our example is the phrase 一時 ("at one time"). In the unfiltered results, we see that for T150A, all five witnesses (CB = the CBETA version, 元 = the Yuan edition of the canon, 宋 = the Song edition, 明 = Ming, 大 = the Taishō) share the same number of instances of 一時 (47 instances). But in the unfiltered results, this same information is repeated on five different rows, one for each witness.

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T
1	ngram	size	work	sig	n	count	label													
2	一時	2	T0150A	CB		47	T150A													
3	一時	2	T0150A	元		47	T150A													
4	一時	2	T0150A	大		47	T150A													
5	一時	2	T0150A	宋		47	T150A													
6	一時	2	T0150A	明		47	T150A													
7	一時	2	T0735	CB		1	T735													
8	一時	2	T0735	元		1	T735													
9	一時	2	T0735	大		1	T735													
10	一時	2	T0735	宋		1	T735													
11	一時	2	T0735	高		1	T735													
12	一時	2	T0735	明		1	T735													
13	一時	2	T0735	麗-CB		1	T735													
14	一時佛	3	T0150A	CB		47	T150A													
15	一時佛	3	T0150A	元		47	T150A													
16	一時佛	3	T0150A	大		47	T150A													
17	一時佛	3	T0150A	宋		47	T150A													
18	一時佛	3	T0150A	明		47	T150A													
19	一時佛	3	T0735	CB		1	T735													
20	一時佛	3	T0735	元		1	T735													
21	一時佛	3	T0735	大		1	T735													
22	一時佛	3	T0735	宋		1	T735													

By contrast, in the filtered version ("default filters, sorted..."), there is only a single entry representing this same information about 一時 in T150A. The different witnesses for each text are sorted into one row together, since for all witnesses, the same n-gram has the same count:



	A	B	C	D	E	F	G	H
1	ngram	size	work	sigla	count	label	label count	
71	思想何等為思想習識	9	T0150A	明,明,明	0	T150A	1	
72	生死識耳鼻口身意	8	T0150A	CB,大	1	T150A	1	
73	生死識耳鼻口身意	8	T0150A	明,明,明	0	T150A	1	
74	思想耳鼻口身意	7	T0150A	CB,大	1	T150A	1	
75	思想耳鼻口身意	7	T0150A	明,明,明	0	T150A	1	
76	聞如是一時佛在	7	T0150A	CB,元,大,宋,明	47	T150A	47	
77	聞如是一時佛在	7	T0735	CB,元,大,宋,宮,明,羈-CB	1	T735	1	
78	講何等為思想盡	7	T0735	CB,元,大,宋,宮,明,羈-CB	1	T735	1	
79	佛說如是比丘	6	T0150A	CB,元,大,宋,明	2	T150A	2	
80	得道佛說如是	6	T0150A	CB,元,大,宋,明	1	T150A	1	
81	結無有結意脫	6	T0150A	CB,元,大,宋,明	1	T150A	1	
82	苦轉法如是為	6	T0150A	CB,元,大,宋,明	1	T150A	1	
83	諦定為八如是	6	T0150A	CB,元,大,宋,明	1	T150A	1	
84	講何等為生死	6	T0150A	CB,元,大,宋,明	1	T150A	1	
85	識耳鼻口身意	6	T0150A	CB,大	1	T150A	1	
86	識耳鼻口身意	6	T0150A	明,明,明	0	T150A	1	
87	耳鼻口身意裁	6	T0150A	元,宋,明	1	T150A	1	
88	耳鼻口身意裁	6	T0150A	大,大	0	T150A	1	
89	講盡受行為講	6	T0150A	元,宋,明	1	T150A	1	
90	講盡受行為講	6	T0150A	大,大	0	T150A	1	
91	如是為講盡	5	T0150A	CB,元,大,宋,明	1	T150A	1	

Similarly, for all Difference and Intersect results, Filter/Rationalize adds zero counts for witnesses in which a given n-gram does not occur at all, if the same n-gram does occur in some other witness of the same work. The reason for this step is this: A TACL database obviously includes counts only for n-grams which *do* occur. With raw results, for which zero counts have not been added, users would have to infer the absence of the n-gram from the silence of the evidence, which might require superhuman powers of observation and attention. In the case of our example, as mentioned above in the Intersect section, there is a variant reading in the Taishō witness of T150A, which breaks the 79-gram otherwise found by the Intersect test between T150A and T735. Thanks to the addition of zero counts, however, we can more easily notice that the 79-gram is missing in the Taishō witness, and thereby, be made aware of the variant reading:

ngram	size	work	sigla	count	label	label count
生死雷讀何等為生死雷讀裁畫為生死雷讀何等為生死雷讀受行讀為是八行讀諦見至諦定為八如是為生死欲滅受行讀	79	T0150A	元,宋,明	1	T150A	1
生死雷讀何等為生死雷讀裁畫為生死雷讀何等為生死雷讀受行讀為是八行讀諦見至諦定為八如是為生死欲滅受行讀	79	T0735	CB,元,大,宋,宮,明,麗-CB	1	T735	1
生死雷讀何等為生死雷讀裁畫為生死雷讀何等為生死雷讀受行讀為是八行讀諦見至諦定為八如是為生死欲滅受行讀	79	T0150A	大,大	0	T150A	1
讀何等為思想要讀所思想欲食能解欲食能斷欲食能自度如是為思想要讀何等為生死讀為六身生死識眼裁生死識耳鼻	69	T0150A	元,宋,明	1	T150A	1
讀何等為思想要讀所思想欲食能解欲食能斷欲食能自度如是為思想要讀何等為生死讀為六身生死識眼裁生死識耳鼻	69	T0735	CB,元,大,宋,宮,明,麗-CB	1	T735	1
讀何等為思想要讀所思想欲食能解欲食能斷欲食能自度如是為思想要讀何等為生死讀為六身生死識眼裁生死識耳鼻	69	T0150A	大,大	0	T150A	1
畫為生死雷讀何等為生死雷讀受行讀為是八行讀諦見至諦定為八如是為生死欲滅受行讀何等為生死味讀所為生死因	67	T0150A	CB,大	1	T150A	1
畫為生死雷讀何等為生死雷讀受行讀為是八行讀諦見至諦定為八如是為生死欲滅受行讀何等為生死味讀所為生死因	67	T0150A	明,明,明	0	T150A	1
生死要讀所為生死欲食隨欲食能斷欲食能如是為生死要讀何等為讀身六衰讀眼裁讀耳鼻口身意裁讀如是為讀識何等	66	T0150A	元,宋,明	1	T150A	1
生死要讀所為生死欲食隨欲食能斷欲食能如是為生死要讀何等為讀身六衰讀眼裁讀耳鼻口身意裁讀如是為讀識何等	66	T0735	CB,元,大,宋,宮,明,麗-CB	1	T735	1
生死要讀所為生死欲食隨欲食能斷欲食能如是為生死要讀何等為讀身六衰讀眼裁讀耳鼻口身意裁讀如是為讀識何等	66	T0150A	大,大	0	T150A	1
何等為讀畫受行為讀八行諦見至諦定為八如是為讀畫欲受行如諦讀何等為讀味知所讀因緣故生樂生喜意如是為味生	58	T0150A	CB,元,大,宋,明	1	T150A	1
何等為讀畫受行為讀八行諦見至諦定為八如是為讀畫欲受行如諦讀何等為讀味知所讀因緣故生樂生喜意如是為味生	58	T0735	CB,元,大,宋,宮,明,麗-CB	1	T735	1
能度如是為要讀如是比丘七處為覺知何等為七色習畫道味苦要是五陰各有七事何等為三觀讀亦有七事得五陰成六衰	53	T0150A	CB,元,大,宋,明	1	T150A	1
能度如是為要讀如是比丘七處為覺知何等為七色習畫道味苦要是五陰各有七事何等為三觀讀亦有七事得五陰成六衰	53	T0735	CB,元,大,宋,宮,明,麗-CB	1	T735	1
讀何等為思想要讀所思想欲食能解欲食能斷欲食能自度如是為思想要讀何等為生死讀為六身生死識眼	44	T0150A	CB,大	1	T150A	1
讀何等為思想要讀所思想欲食能解欲食能斷欲食能自度如是為思想要讀何等為生死讀為六身生死識眼	44	T0150A	明,明,明	0	T150A	1
讀讀諦見到諦定意為八如是畫思想受行讀何等為思想味讀所為思想因緣生樂得意	35	T0150A	CB,大,明	1	T150A	1
讀讀諦見到諦定意為八如是畫思想受行讀何等為思想味讀所為思想因緣生樂得意	35	T0735	CB,元,大,宋,宮,明,麗-CB	1	T735	1
讀讀諦見到諦定意為八如是畫思想受行讀何等為思想味讀所為思想因緣生樂得意	35	T0150A	宋,宋	0	T150A	1

These first two "tidying" operations—grouping all witnesses together where they have the same counts, and adding explicit indication where an n-gram is missing in a witness—are achieved in the background for Difference and Intersect tests alike. Intersect results alone undergo two additional operations. Filter/Rationalize works in the background to "extend" all strings found in the Intersect results out to maximum length. Recall that one of the settings for a database is maximum n-gram length, for which we have recommended 8-grams. "Extend" is the function that allows us, despite this limitation, to find very long matching strings in an Intersect test, like the 79-gram seen in the screenshot above. Since the primary function of Difference is to uncover distinctive features of style, which by definition must recur, it is usually not helpful to find long strings unique to a work. For this reason, "extend" is only applied to Intersect, but not to Difference.

Intersect results also undergo one more tidying operation in Filter/Rationalize. Inevitably, the longer strings resulted from the "extend" operation just described comprise numerous shorter strings, and those shorter strings are also included in the raw results file, since they also meet the condition of being shared by the texts under comparison (included in the intersection of the two texts). We can see this clearly in this little snippet from the unfiltered results:

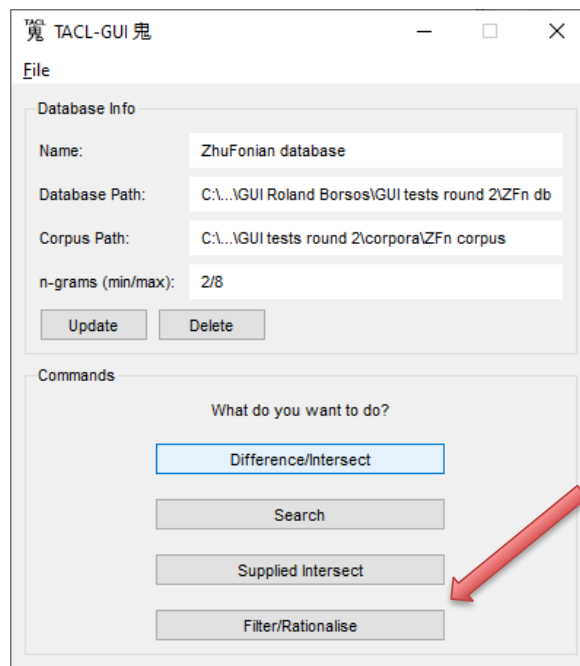
	A	B	C	D	E	F	G
1	ngram	size	work	siglum	count	label	
2101	聞佛	2	T0150A	CB	4	T150A	
2102	聞佛說	3	T0150A	CB	1	T150A	
2103	聞如	2	T0150A	CB	47	T150A	
2104	聞如是	3	T0150A	CB	47	T150A	
2105	聞如是一	4	T0150A	CB	47	T150A	
2106	聞如是一時	5	T0150A	CB	47	T150A	
2107	聞如是一時佛	6	T0150A	CB	47	T150A	
2108	聞如是一時佛在	7	T0150A	CB	47	T150A	
2109	聞經	2	T0150A	CB	4	T150A	

However, the counts for all these "sub-strings" included in the longer string are the same as counts for the longer string (there are 47 instances each of 聞如, and 聞如是, and 聞如是一, and 聞如是一時, and 聞如是一時佛, and 聞如是一時佛在). The information about the sub-strings therefore teaches the human user nothing about the relation between texts that is not already known from the result for the main, longer string. Filter/Rationalize therefore removes any shorter strings that are part of a longer string, so long as they have the same count. For this reason, in the "filtered" results, there is simply no longer any result for 一時 alone (you can search the spreadsheet and see). Thanks to the "extend" operation, all redundant strings have been removed, and therefore, all the original entries containing 一時 have been extended and condensed into the two entries for 聞如是一時佛在 (one each for T150A and T735).

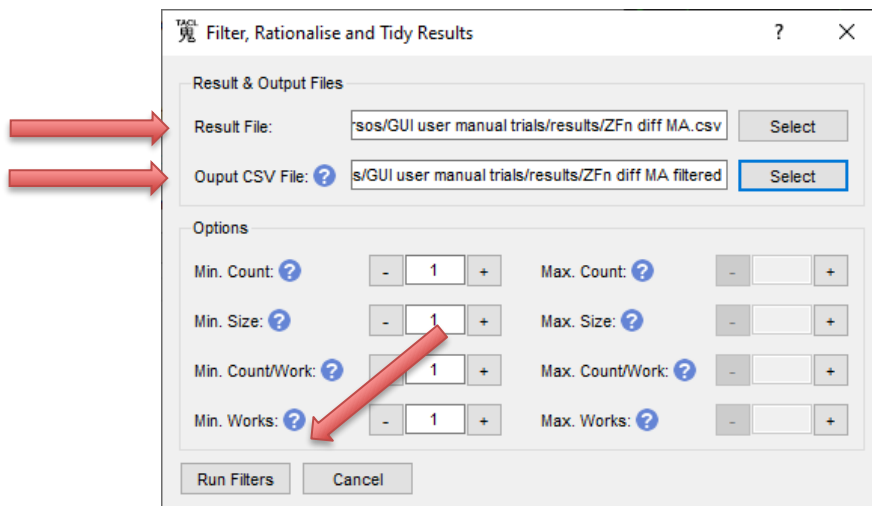
We recommend that users *always* run Difference and Intersect results through "Filter/Rationalize" at the very last step before they import and view the results in Excel, in order to take advantage of these "tidying up" functions. Results will otherwise be far messier and more redundant, and take much longer to work through.

However, again, always Filter/Rationalize results as the very last step only, before you move to human analysis of the raw results.

With these preliminaries in mind, implementing the most basic "tidy-up" Filter/Rationalize operation is simple. At the TACL GUI main menu, click "Filter/Rationalize".

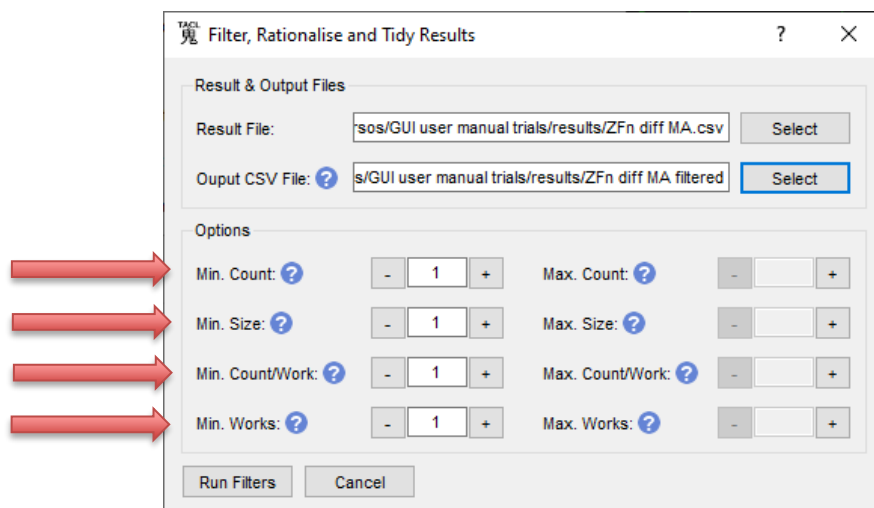


You then get this screen:



As in the example seen in this screenshot, in the "Result File" box, select a raw results file to be processed. In the "Output CSV File", stipulate a location and filename for the Filter/Rationalized results. Then click "Run Filters". You will get a set of results in which redundancies have been eliminated, witnesses with the same results have been grouped on a single row of the table, information has been added for witnesses in which the count of a given n-gram is zero, and (if your results are Intersect results), results have been "extended" to find very long n-grams that match the search conditions.

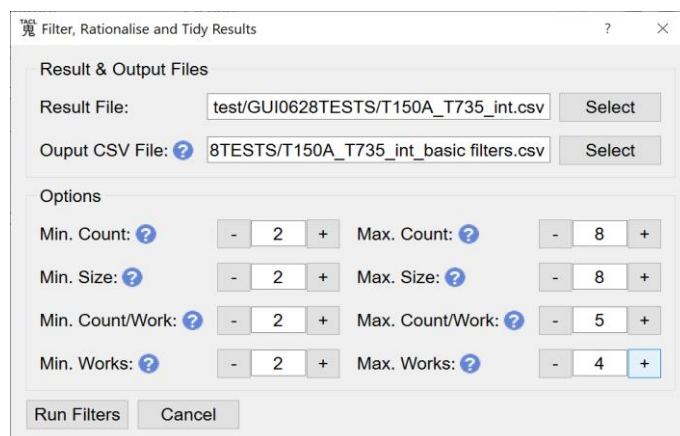
In addition to these default “tidying” operations, however, as already mentioned, and as can be guessed from the screenshot above, the Filter/Rationalize function also allows users to apply various parameters to filter results:



- Min(imum) Count/Max(imum) Count: allows you to specify a lowest or highest count, as a condition for an n-gram to be included in the filtered results. "Count" means here the number of times that an n-gram appears in a single group of texts identified by a label. For example, select min: 3 and max: 7 to keep only n-grams that occur 3-7 times in all the results combined (regardless of distribution across individual works).
- Min Size/Max Size: include or exclude n-grams, depending upon their length. "Size" here means the length of the n-gram (the number of characters = tokens). For example, select min: 4 to keep only 4-grams and longer; select max: 6 to keep only 6-grams and shorter.
- Min Count/Work and Max Count/Work: include or exclude n-grams depending on how many times they occur in *every work* in the test. For example, select min: 2 to keep only n-grams that occur at least 2 times in every work.
- Min Works/Max Works: include or exclude n-grams depending upon how many works they occur in within a single group of texts identified by a label, independent of the number of occurrences in each work. For example, select min: 3 to keep only n-grams that occur in at least 3 works.

These options can be useful, because TACL tests on large texts or corpora can often yield raw results so copious that the human analyst is overwhelmed, or does not know where to start. In fact, it is sometimes the case that raw results can be so copious that they also overwhelm Excel (and make it crash), in which case, it might not just be advisable to run some filters, but necessary. Here, then, filters allow us to isolate some sub-set of the initial results, according to criteria we guess enhance our likelihood of efficiently finding what we are looking for.

A screen in which a user makes fairly full use of these various options might look like this:



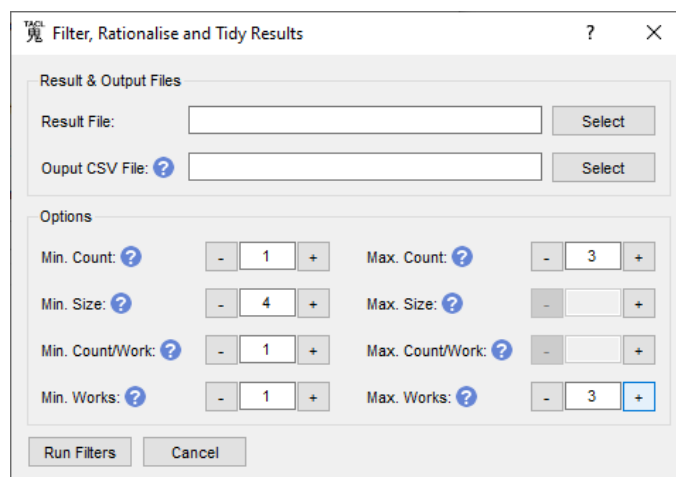
The decision about what parameters to use in filtering depends upon such factors as the type of test one is conducting, what one is looking for, and the quantity of raw results that one has to work through. In our experience, we learn more about how to use these filter parameters through trial and error. However, presuming that Difference tests are being applied to discover style, and Intersect tests to discover sources of a text, it is possible to recommend some general guidelines for each type of test—at least as a starting point for further experimentation.

#### 1) Intersect:

If the main purpose of an Intersect test is to uncover debts of one text to another text or texts, we would, generally speaking, be more interested in longer strings, which are only found in a few works. Shorter strings are more likely to be common vocabulary or stock phrases; the same is true of strings found in many works. The limit case is the unique match, which only occurs once each in precisely two texts in the entire canon (an example is the 76-gram already shown above, p. 46). Such items can often be "smoking guns", which show beyond doubt that one text has borrowed from another.

On this basis, for Intersect tests, we recommend starting with the following parameters (and then tweaking, if you find it leaves you with too many results, or too few):

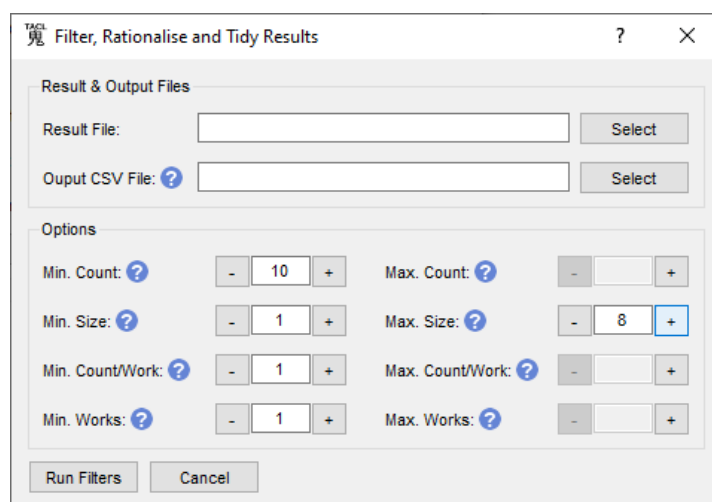
- Min. size: 4
- Max. count: 3
- Max. works: 3



## 2) Difference:

The main purpose of a Difference test, on the other hand, is to uncover stylistic features of a translator. By definition, traits we treat as evidence of style recur—indeed, all other things being equal, the more often they recur in a given author, the stronger they appear as a trait of that author. Such recurring items are usually relatively short (single words, or short collocations and phrases). In this case, then, longer strings are less likely to be useful evidence, as are things that occur fewer times. Therefore, for typical Difference tests, we recommend starting with the following parameters:

- Max. size: 6-8
- Min. count: 10



Min count, however, especially requires adjustment, depending on corpus size. In analysis of a very large corpus, you may have too many results with a min count of 10, and need to adjust upwards; for a small corpus, you might need to adjust downwards.

### 3) Supplied Intersect

Generally speaking, Supplied Intersect tests will take as input the raw results of at least one Difference test, and such tests are usually used for the same ends as typical Difference tests (discovery of author or translator style). For these reasons, we recommend roughly the same Filter/Rationalize parameters for Supplied Intersect as for Difference.

Note also that filtering by size and counts works in conjunction with the Excel sort protocols that we will recommend below. This means that it is not absolutely necessary to get your filters exactly "right" before you start human analysis—you can often compensate for results that are too copious by looking at only parts of a sorted set of raw results in Excel. Note, also, that if you find your initial filter settings yielded too many results, or too few, you can always go back to the TACL GUI and run a new Filter/Rationalize operation that is more stringent, or more relaxed.

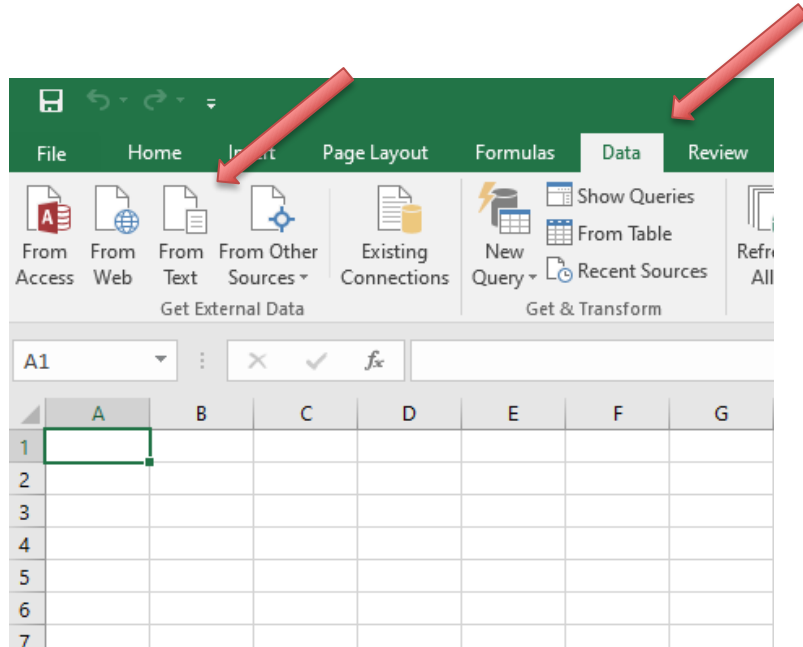
#### **WORKFLOW STEP 7: Working with the results (1): Import and sorting in Excel (or similar)**

The TACL GUI outputs its results in the form of .csv files ("[comma separated values](#)"). A simple way of working with .csv results is to use Excel (or something comparable, such as [LibreOffice](#); or Calc from [Apache OpenOffice](#); or [WPS](#), which reportedly is found good by Chinese users). More advanced users may have their own ways of handling the .csv files, but we will here walk users through the process in Excel.

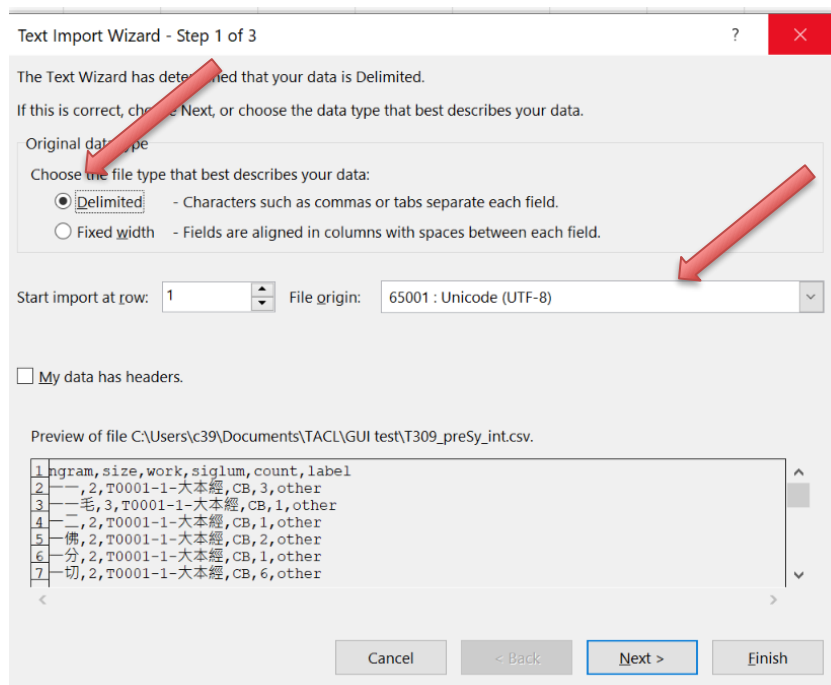
Please note that all our examples here were prepared using Excel 2016. If your version of Excel differs, the layout of your screen may also be slightly different from what we describe and show.

In Windows, where Excel is set as the default application for .csv files, it is unfortunately not possible to correctly open a .csv results file simply by clicking on it—the file will display as garbled junk. Rather, open Excel, go to the “Data” tab in the menu bar at the top of the page, click “From Text”, and navigate to your .csv file.

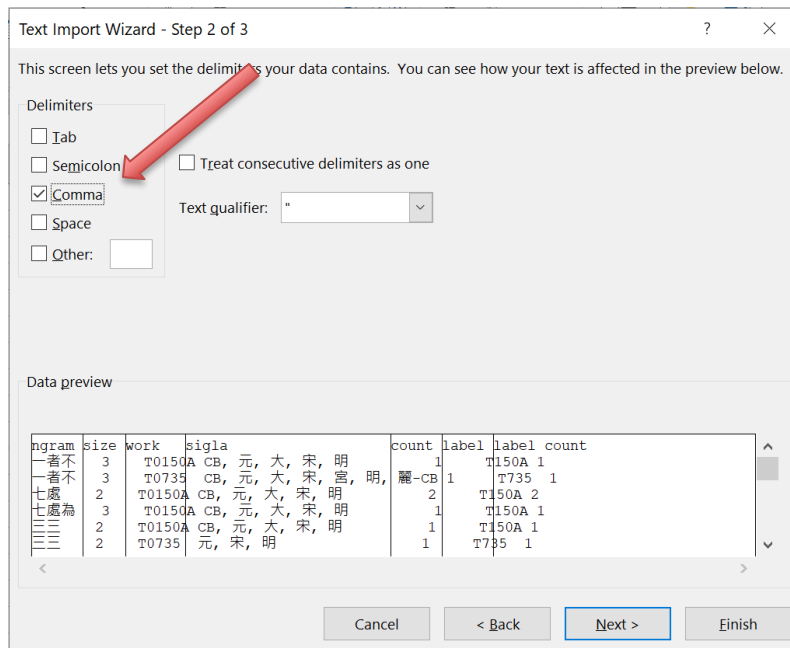




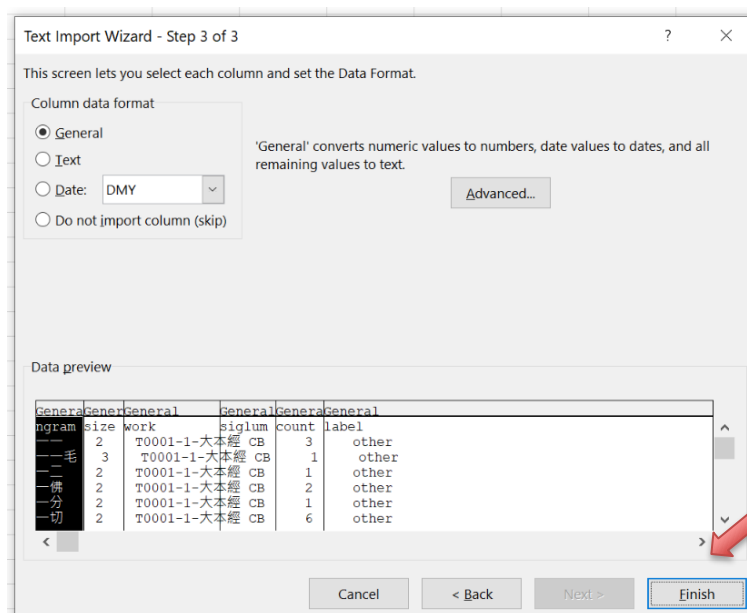
Next, you will see a dialogue window for “Text Import Wizard” pop up:



Under “original data type”, select “Delimited”. For “File origin”, make sure to select “65001: Unicode (UTF-8)”, to ensure proper display of the Chinese characters—otherwise, you will just get gobbledy-gook. Then click “Next”.

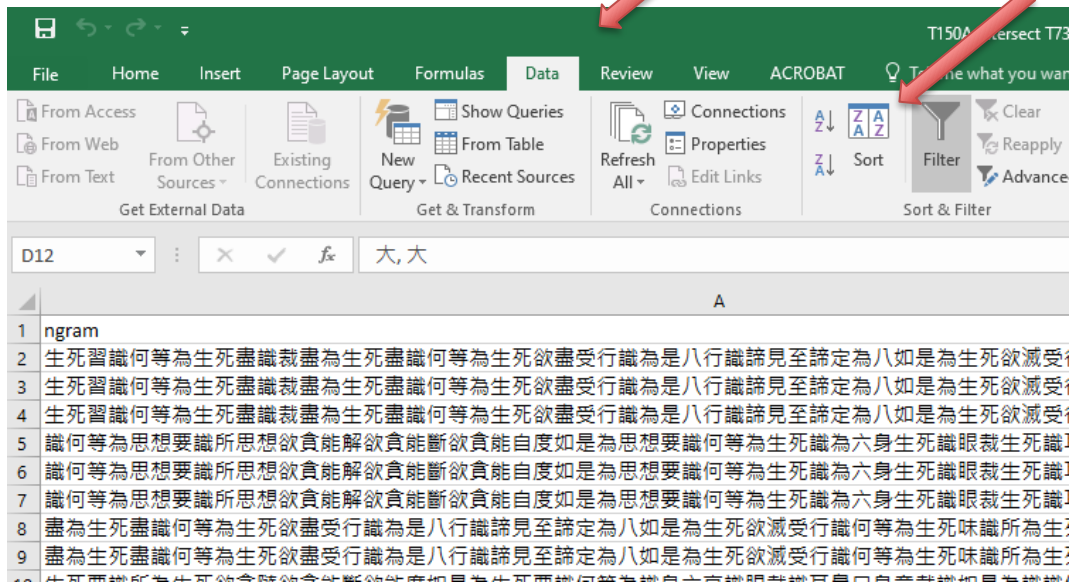


Under “Delimiters”, select “Comma”. Then click “Next”.

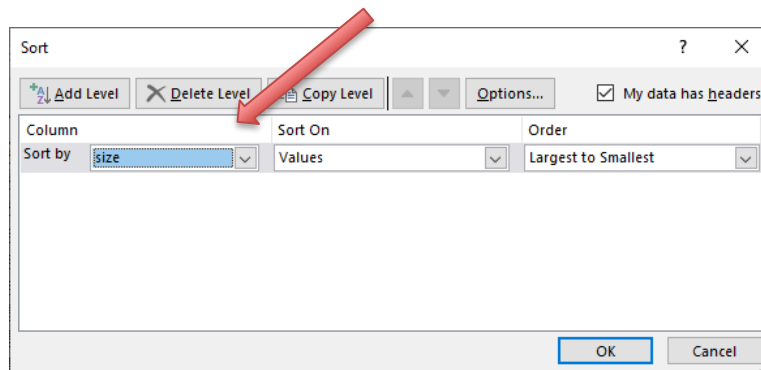


Finally, simply click “Finish”.

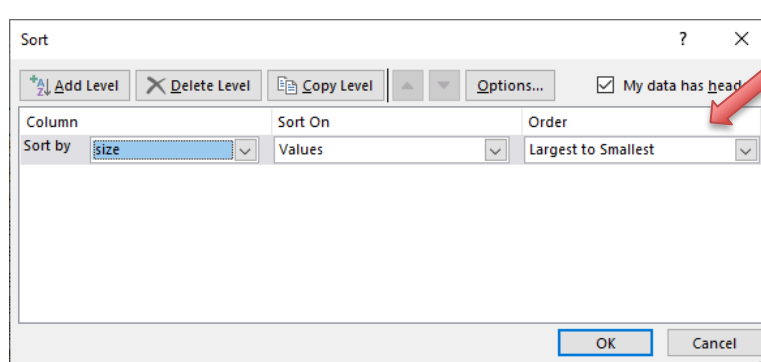
Once you have your results imported to Excel, you can make your own work analysing the results much more efficient by taking advantage of the "Sort" function. Sort is found in the "Data" tab:



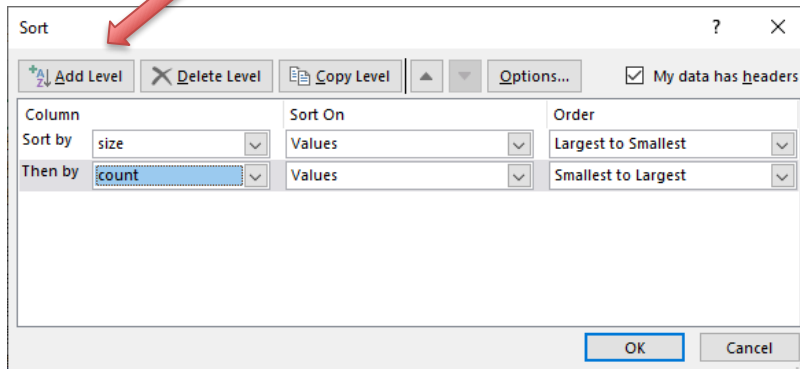
In "Sort" you can select any column header (e.g. n-gram, work, size, count), and nominate it as the criterion for the sort.



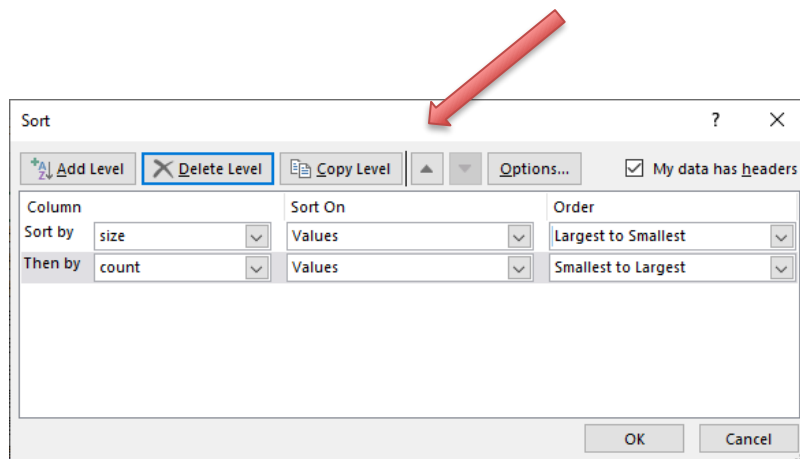
On the right, you can dictate whether the ordering will be in ascending or descending order of size (or, where relevant, forward or backward alphabetical order, etc.).



You can also add levels, which means that you sort first by the top criterion, and then, within results grouped together by that sort, by the second, and so on.



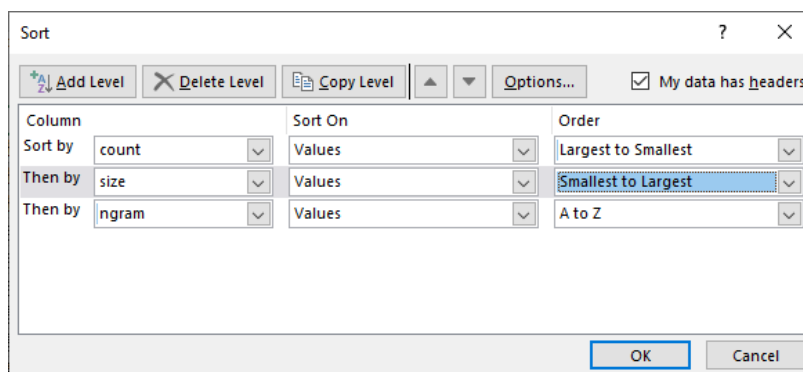
If you want to shuffle levels, you can use the "up" and "down" arrows:



We recommend the following default sort protocols, depending upon the type of TACL test results you are analysing:

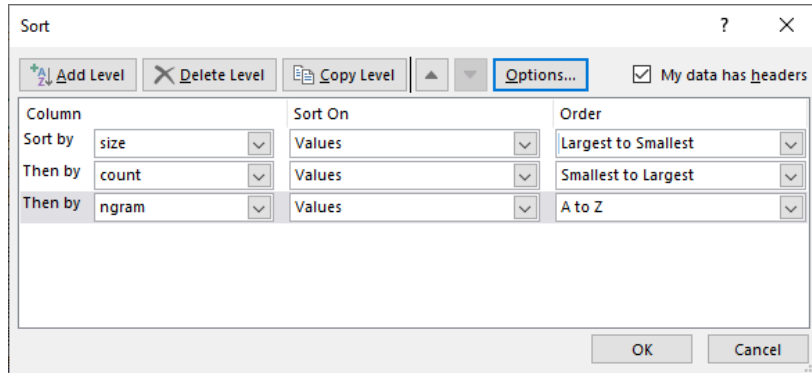
Difference:

- count, descending (largest to smallest)
- size, ascending (smallest to largest)
- n-gram



Intersect (including Supplied Intersect):

- size, descending (largest to smallest)
- count, ascending (smallest to largest)
- n-gram



Search: Sort on "number of n-grams", in descending order ("Largest to Smallest").

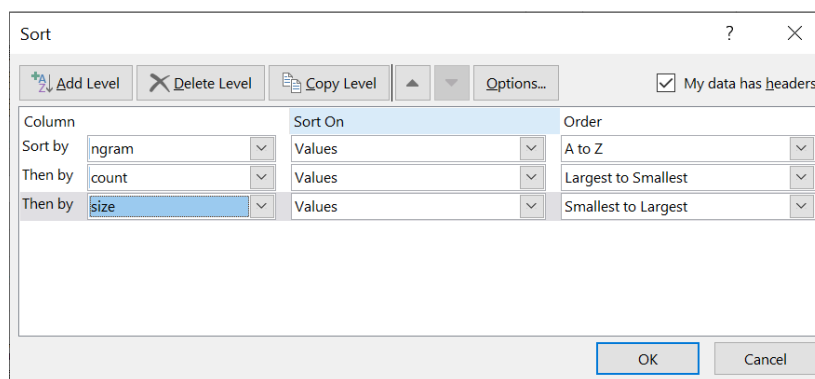
Finally, we here introduce one more sorting trick in Excel, which can increase our efficiency in browsing results to find items most likely of evidential interest.

When we use the sort protocols describe above, we occasionally run into the problem that multiple entries for the same individual n-gram can be widely scattered in different parts of our results. This problem can be addressed by a slightly more complex sorting protocol, which preserves the sort criteria laid out above, but still groups together all entries for the same n-gram. We take as our example the results of the Supplied Intersect test (“Ekottarikagama intersect ZhuFonian diff Madhyamagama”, i.e. EXAMPLE 1). If we simply sort by count in descending order, we see the following:

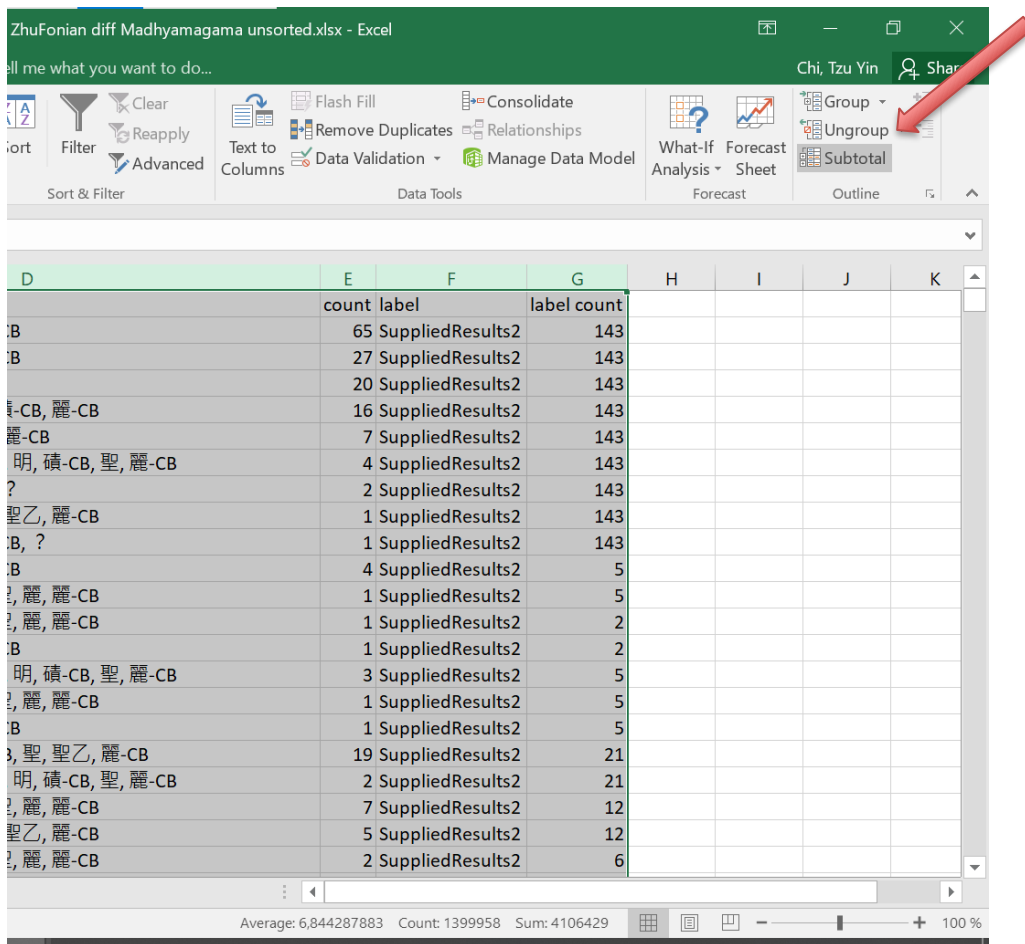
	A	B	C	D	E	F	G
1	ngram	size	work	sigla	count	label	label count
2	羯磨	2	T1428	元, 未, 宮	2115	SuppliedResults2	2117
3	羯磨	2	T1428	明	2110	SuppliedResults2	2117
4	羯磨	2	T1428	聖	2103	SuppliedResults2	2117
5	羯磨	2	T1428	CB, 大, 磧-CB, 麗-CB	2102	SuppliedResults2	2117
6	羯磨	2	T1428	聖乙	2100	SuppliedResults2	2117
7	群比	2	T1428	CB, 大, 磧-CB, 麗-CB	1016	SuppliedResults2	1207
8	群比	2	T1428	元, 未, 明	1011	SuppliedResults2	1207
9	知智	2	T1543	元, 未, 明, 聖乙	1010	SuppliedResults2	1051
10	知智	2	T1543	CB, 南藏, 大, 宮, 磧-CB, 麗-CB	1009	SuppliedResults2	1051
11	群比	2	T1428	宮	1009	SuppliedResults2	1207
12	知智	2	T1543	聖	1008	SuppliedResults2	1051
13	群比	2	T1428	聖乙	1008	SuppliedResults2	1207
14	群比	2	T1428	聖	1006	SuppliedResults2	1207
15	現在前	3	T1543	CB, 南藏, 大, 磧-CB, 麗-CB	864	SuppliedResults2	1020
16	現在前	3	T1543	聖乙	862	SuppliedResults2	1020

Entries for 群比 are separated from one another because we have ordered first by count, but these entries do not share the same count, and other n-grams with counts in the same range intrude between them (in fact, entries for other n-grams are also scattered, but they fall beyond this screenshot).

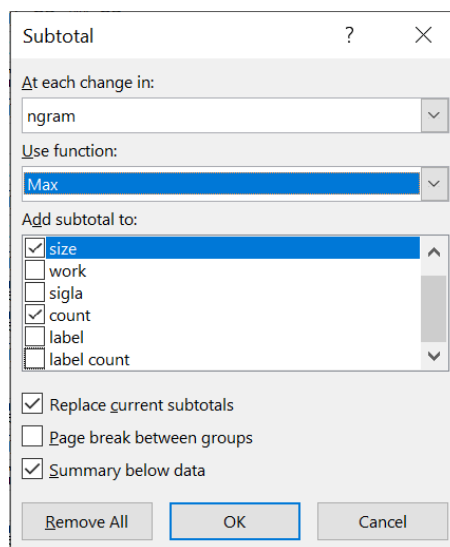
To address this problem, we recommend the following sorting protocol. First, sort by n-gram, then (as usual for Difference tests) count (in descending order), and then (optionally) size (in ascending order).



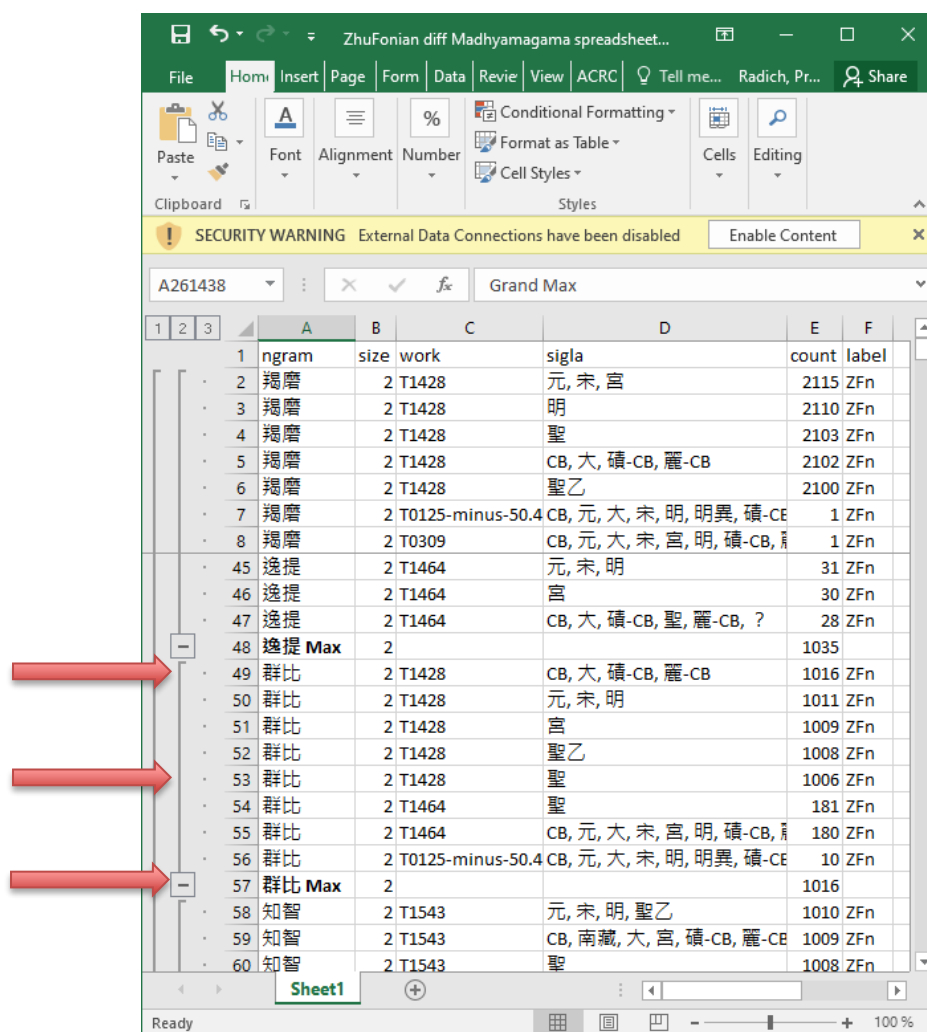
Then select all data, go to the “Data” tab, and at the far right end of the toolbar, click “Subtotal”.



In the Subtotal window, for “At each change in”, select “ngram”; for “Use Function, select “Max” or "Sum", depending on your priorities in analysing the evidence—"Max" will produce a listing in order of the *maximum count* for each n-gram; "Sum" in order of the *total count*. Then, under “Add subtotal to”, select "count". Leave ticked the defaults "Replace current subtotals" and "Summary below data". Click “OK”.



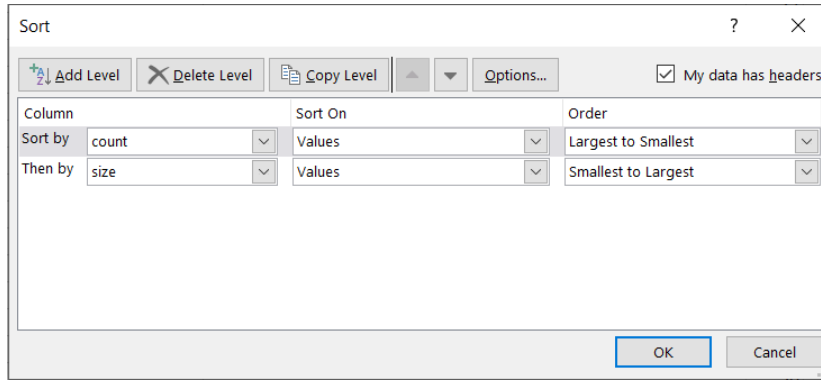
Once you click “OK”, Excel will start re-sorting. The sort may take a while if you have a lot of data. Eventually, when it finishes, it should look like this:



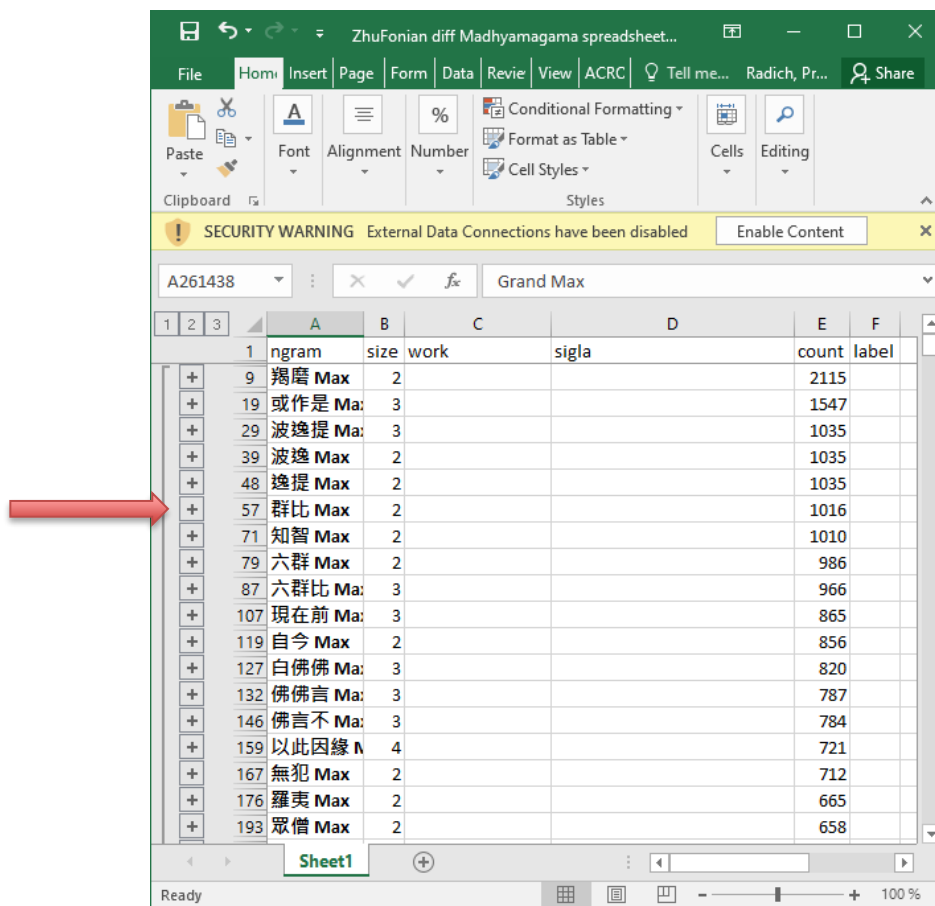
All the entries for the same n-gram are now grouped together.

Now, click on “2” at the top left corner of the spreadsheet to collapse each n-gram subgroup, and “Sort” again by count in descending order. In the dialogue window, you will see the sorting scheme you first used prior to the subtotal operation (see above p. 77). Remove the "ngram" level and, leave the "count" and "size" (though again, "size" is optional). The resulting sorting will now follow the “Max” or “Sum” count, depending on which one you chose earlier.





Now, not only are all the entries for the same n-gram grouped together, but the subgroups are neatly sorted in descending order of their maximum/total count. You can then browse more entries at one time:



To examine the individual entries under any subgroup, just click on the plus sign (+) under 2 to expand the entry.

The screenshot shows an Excel spreadsheet with the following data:

	A	B	C	D	E	F
1	ngram	size	work	sigla	count	label
9	羯磨 Max	2			2115	
19	或作是 Ma:	3			1547	
29	波逸提 Ma:	3			1035	
39	波逸 Max	2			1035	
48	逸提 Max	2			1035	
49	群比	2	T1428	CB, 大, 磧-CB, 麗-CB	1016	ZFn
50	群比	2	T1428	元, 宋, 明	1011	ZFn
51	群比	2	T1428	宮	1009	ZFn
52	群比	2	T1428	聖乙	1008	ZFn
53	群比	2	T1428	聖	1006	ZFn
54	群比	2	T1464	聖	181	ZFn
55	群比	2	T1464	CB, 元, 大, 宋, 宮, 明, 磧-CB, 麗	180	ZFn
56	群比	2	T0125-minus-50.4	CB, 元, 大, 宋, 明, 明異, 磧-CE	10	ZFn
57	群比 Max	2			1016	
71	知智 Max	2			1010	
79	六群 Max	2			986	
87	六群比 Ma:	3			966	
107	現在前 Ma:	3			865	

**WORKFLOW STEP 8: Working with the results (2): Examining results back in context**

Finally, it bears emphasising once more that raw TACL results are usually not yet "evidence" for the purpose of typical research questions. They usually still require analysis by a philologically expert human, with reference to the meaning and contexts in which the strings appear in our texts. (It is an open research question in its own right, whether raw TACL results could be used to build solid arguments without an intervening step of such qualitative human analysis.) To return briefly to an example already given above, in a certain test on Lokakṣema's *Kāśyapa-parivarta* T350, we see the following item among our results:

	A	B	C	D	E	F
1	ngram	size	work	sigla	count	label
2	譬如	2	T0350	元, 未, 明, 聖	69	T350
3	譬如	2	T0350	CB, 大, 宮, 麗-CB	68	T350
4	是為	2	T0350	CB, 元, 大, 未, 宮, 明, 聖, 麗-CB	50	T350
5	迦葉	2	T0350	CB, 元, 大, 未, 宮, 明, 聖, 麗-CB	47	T350
6	比丘	2	T0350	CB, 元, 大, 未, 宮, 明, 聖, 麗-CB	38	T350
7	五百	2	T0350	CB, 元, 大, 未, 宮, 明, 聖, 麗-CB	36	T350
8	沙門	2	T0350	CB, 元, 大, 未, 宮, 明, 聖, 麗-CB	33	T350
9	為四	2	T0350	CB, 元, 大, 未, 宮, 明, 聖, 麗-CB	32	T350
10	佛語	2	T0350	元, 未, 宮, 明, 聖	30	T350
11	一者	2	T0350	CB, 元, 大, 未, 宮, 明, 聖, 麗-CB	29	T350
12	二者	2	T0350	CB, 元, 大, 未, 宮, 明, 聖, 麗-CB	29	T350
13	佛語	2	T0350	CB, 大, 麗-CB	29	T350
14	四事	2	T0350	CB, 元, 大, 未, 宮, 明, 聖, 麗-CB	27	T350
15	語迦	2	T0350	元, 未, 宮, 明, 聖	27	T350

meaning that 佛語 appears 29 or 30 times in the results (depending upon the witness for T350) (the test was a Difference, trying to find stylistic traits that might distinguish Lokakṣema from another point of comparison. When we see the results in isolation like this, it might be easy to think that 佛語 is *fóyǔ*, and means "the word of the Buddha" (at least, that is what I thought when I first saw it). However, in context:

CBETA CBReader 2011 - [佛說遺日摩尼寶經 [卷1] -- T12, No. 0350]

Text Search: 佛語迦葉

Found: 70, Time: 1 sec

Fou...	T...	V...	Sutr...	Sutra Title	F...	Category	Eyline
1	T	03	0185	太子瑞應本起經	2	本緣部	【吳支謙譯】
1	T	03	0186	普曜經	8	本緣部	【西晉竺法護譯】
4	T	03	0187	方廣大莊嚴經	12	本緣部	【西地婆訶羅譯】
1	T	09	0263	正法華經	3	法華部	【西晉竺法護譯】
1	T	11	0310	大寶積經	1...	寶積部	【唐善提流志譯】
1	T	12	0345	慧上菩薩問大善權經	1	寶積部	【西晉竺法護譯】
26	T	12	0350	佛說遺日摩尼寶經	1	寶積部	【後漢支婁迦讖譯】
5	T	22	1421	彌沙塞部和鞠五分律	16	律部	【劉宋佛陀什共竺】
4	T	24	1469	佛說迦葉藥戒經	1	律部	【宋沮渠京聲譯】
7	T	26	1523	大寶積經論	1	釋經論部	【後魏菩提流支譯】
2	T	26	1523	大寶積經論	2	釋經論部	【後魏菩提流支譯】
2	T	26	1523	大寶積經論	3	釋經論部	【後魏菩提流支譯】
1	T	31	1610	佛華嚴	?	持世部	【後魏菩提流支譯】

道。譬如郡國多積糞壤。有益稻田菜園。菩薩雖在愛欲中。益於天上天下。佛語迦葉。若有菩薩欲學極大珍寶之積遺日羅經。當隨是經本法精進。何等為本法。無法無我無人無壽無常無色無痛痒無思想無生死識。是為法本根。有常在一邊。無常在一邊。有常無常適在其中。無色無見無識。是故為中之智點本也。譬如大地為一界。復一佛界。兩界之際中。無色無見無識無我無識無所入無所語。是為智點本也。心為一邊。無心為[20]一邊。設無心無識無我無識。是為中間之本。諸佛經法等。無有異有德無德。內事外事。有世間無世間。為度者未度者。脫愛欲未脫愛欲。泥洹等無有異。有在一邊。無有在一邊。有無有適在中間。是為智點中本也。佛語迦葉。我為汝曹說法。從生至死身所出生。苦癡在一邊。點在一邊。無癡無點適在中間。是為智點中間之本。

We see that in fact, 佛語 is *Fó yǔ*..., "the Buddha said to [Kāśyapa]". It turns out that this is true of all instances of 佛語 in T350. Now, for the purposes of distinguishing translation style, it might be

important to distinguish between *fóyǔ* and *Fó yù*—but since these two words/collocations are graphically identical, TACL cannot tell the difference. Only the human can do this work.

We describe some considerations that we must bear in mind at this stage in a little more detail in the "TACL Methods Guide".

### **Beyond the GUI**

A little supplementary information for technically minded users, or users who have caught the bug through the GUI, and are interested in going further: TACL was originally programmed for use from the command line, and the command-line version of the code provides users access to a wider and more flexible range of functions than the TACL GUI. The TACL code can also be accessed via an API, and called by other code, which means that an additional level of power again, beyond direct use at the command-line, can be achieved by writing programmes that coordinate many TACL operations in larger operations. The underlying TACL code is programmed in Python. It is free software, meaning that users can access all details of the code, and are free, both technically and legally, to modify it and use it in any way they wish. The full TACL code is always available and up to date on [GitHub](#). Documentation for TACL (the underlying TACL code, *not* this GUI) can be found [here](#).

### **Glossary**

Listed in alphabetical order. Terms italicised within each entry are explained in other Glossary entries.

asymmetric Difference: See "Difference". See also "Asymmetric Difference", p. 49.

catalogue: A text-only file which is used to tell TACL which text groups or *corpora* individual works belong too. Each line in a catalogue file comprises two elements: an identifier for a work in the *corpus*, and a *label*. See WORKFLOW STEP 3: "Catalogues", p. 33.

concatenated test: A TACL test comprising two or more *Difference* or *Intersect* operations, the results of which are combined. See *Supplied Intersect*.

corpus: A text or group of texts (a one-text corpus is also possible and useful) defined by the user for the purposes of TACL analysis. Corpora are defined for TACL by the use of *labels* in *catalogue* files. Corpora must be in the custom format required for use by TACL. Base corpora are available for download. See WORKFLOW STEP 1: Corpora, p. 19.

.csv: "comma separated values"—the file format in which TACL (and the GUI) outputs results.

Information in the results file is separated into units by commas, which tells the computer where to break between units of information when reading the file—e.g. so that it knows which cells to put the information in when importing to Excel.

database: A database generated by TACL which stores lists of all *n*-grams appearing in each text in the *corpus*, with counts of occurrences for each *n*-gram. Subsequent TACL operations (*Difference*, *Intersect*, *Search*, etc.) are based on the information in the database. See "Here, "元.txt" is the text file containing the witness to T309 from the edition of the canon called "Yuan" by the Taishō editors, as documented by the Taishō apparatus, "大.txt" contains the witness of the Taishō base text, and so on. (The big corpora we have made available for download are also structured in this manner, as you can confirm for yourself by looking directly at the files.)

It is important for users also to note that work identifiers (subdirectory names) and filenames of witness .txt files should not contain spaces or special characters. Using spaces and special characters can create complications at later steps of the process.

The TACL GUI can only process corpora structured in subdirectories containing text files, in this manner. It is particularly important to bear this point in mind if you wish to manually add works to your corpus that are not represented in the CBETA digitisations. You must then create a subdirectory for each work, and within that subdirectory, a text file for each witness (even where you only have one witness).

In principle, it is possible to use TACL with corpora other than the files digitised by CBETA (e.g. users might be interested in working with texts from the Daozang or the *Si ku quan shu*, or digitisations of texts attested only in manuscript form). However, any such additions to the TACL corpus must be structured in the format just described. Another proviso is that texts must be digitised and tokenised in a form that TACL recognises. We recommend that users who are interested in such extensions of their corpora contact us.

WORKFLOW STEP 2: Databases", p. 21 ff.

Difference: A TACL test which finds all *n*-grams unique to one side of a comparison, or unique to each side (i.e. in a given *corpus*, as defined by *labels* in a *catalogue* file). Results can be restricted to one side of a comparison only by specifying that the Difference test should be "asymmetric". See "Difference", p. 47.

"Filter/Rationalize": A set of post-processing operations allowing users to filter raw results, in order to eliminate results that are probably unwanted or irrelevant, and focus on results most likely of interest. In the TACL GUI, "Filter/Rationalize" operations are achieved at a dedicated dialogue screen. See "WORKFLOW STEP 6: Filter/Rationalize:", p. 63.

identifier: A name or abbreviation used to identify a *work* in naming corpus sub-directories, and in TACL *catalogues*. Identifiers are most typically Taishō numbers (or the equivalent in other collections such as the Xuzangjing/Zokuzōkyō), sometimes modified to reflect changes to the structure of the texts and corpus (e.g. in the modified "Radich corpus"). For example, the identifier of the *Qi Fo jing* 七佛經 T2 is "T0002".

Intersect: A TACL test which finds all *n*-grams shared by both (or all) sides of a comparison (i.e. in all *corpora* under analysis, as defined by *labels* in a *catalogue* file). See "Intersect", p. 39.

label: A user-defined identifier of a text group or *corpus*, appearing at the end of a line in a *catalogue*, assigning the work listed on that line to the text group or corpus in question. E.g., in the following line from a *catalogue*,

```
T0309 ZhuFonian
```

"T0309" is the identifier for the *Shi zhu duan jie jing* 十住斷結經 T309 in the TACL corpus, and "ZhuFonian" is a label assigning that work to a *corpus* called "ZhuFonian" (here for "ZhuFonian").

*n*-gram: A "string" (series of tokens, i.e., for us, Chinese characters) of user-defined length *n*. For example, a 2-gram is two characters long (e.g. 如來); an 7-gram is seven characters long (e.g. 無上平等最正覺). In TACL, users must decide a range of *n*-gram lengths to be compiled into the *database*.

"Radich corpus": A customised *corpus* for TACL use, comprising the CBETA digitised files of the Taishō canon, modified to reflect findings in text-historical scholarship. See "WORKFLOW STEP 1: Corpora", p. 19; "(2a) Download the full database for the "Radich corpus" version of the Taishō", p. 28.

Search: A TACL test which finds all *n*-grams (as defined by a list in a text-only file supplied by the user) in all works in a given *corpus* (as defined by a *catalogue*). See "Search", p. 59.

string: A continuous series of tokens (Chinese characters). For our purposes, "string" is synonymous with *n-gram* (see *n-gram*).

Supplied Intersect: A TACL test which discovers the *intersect* of two results files from other TACL operations (*Intersect* or *Difference* tests)—i.e. it finds all *n-grams* found in both inputted sets of results files. See "WORKFLOW STEP 5: Supplied Intersect", p. 55 .

TACL: The suite of software tools that users access via the TACL GUI. See "Background", p. 5 ff.

“T-X corpus”: A TACL-ready *corpus* comprising the texts of the Taishō and Xuzangjing/Zokuzōkyō corpora, in the form each text takes in the original digitisation by CBETA (i.e. without the modifications of the "*Radich corpus*"). See "WORKFLOW STEP 1: Corpora", p. 19.

witness: An individual version of a *work*, where multiple versions differing in various details have been transmitted, and can be subjected to TACL analysis. For example, a single *work* in the Taishō might be represented in a TACL corpus by text files representing the versions found in editions of the canon labelled as "Song", "Yuan", "Ming" etc. by the editors of the Taishō.

"work": A "text", e.g. a single Taishō text associated with a single Taishō number. We use the term "work" to allow us to distinguish between a work and a *witness*. For TACL purposes, it is also possible to treat a single Taishō text as more than one "work"—for example, when we split the *Liu du ji (jing)* 六度集(經) T152 into individual chapters/tales for the purpose of analysis, as in the modified "*Radich corpus*".

### **TACL-based research publications**

We are very keen to keep abreast of any research that is accomplished through the use of TACL, and to keep the following list up to date, and supplement it. We encourage users to [contact us](#) with information about such publications. A directory of PDFs of most of these publications is included in the "TACL GUI starter kit".

Most of the publications authored or co-authored by Radich below can be found at [academia.edu](http://academia.edu).

Funayama Tōru 船山徹 [Chuanshan Che]. “*Da fangbian Fo bao'en jing bianzuan suoyinyong de Hanyi jingdian*” 《大方便佛報恩經》編纂所引用的漢譯經典. Translated by Wang Zhaoguo 王招國. *Fojiao wenxian yanjiu* 佛教文獻研究 2 (2016): 175-202.

- Lin Qian 林乾 and He Shuqun 何书群 [Michael Radich]. "Zhu Fonian suo 'yi' dasheng jingdian de jisuanji fuzhu wenben fenxi yanjiu" 竺佛念所"译"大乘经典的计算机辅助文本分析研究. *Shijie zongjiao wenhua* 世界宗教文化 (2020), no. 6: 16-22.
- Lin Qian and Michael Radich. "A Computer-assisted Analysis of Zhu Fonian's Original Mahāyāna Sutras." *Buddhist Studies Review* 38, no. 2 (2021): 145-168.
- Radich, Michael. "Tibetan Evidence for the Sources of Chapters of the Synoptic *Suvarṇaprabhāsottama-sūtra* T664 Ascribed to Paramārtha". *Buddhist Studies Review* 32, no. 2 (2015): 245-270.
- Radich, Michael. "On the Sources, Style and Authorship of Chapters of the Synoptic *Suvarṇaprabhāsottama-sūtra* T664 Ascribed to Paramārtha (Part 1)." *Annual Report of The International Research Institute for Advanced Buddhology* 17 (2014): 207-244.
- Radich, Michael. "On the *Ekottarikāgama* 增壹阿含經 T 125 as a Work of Zhu Fonian 竺佛念." *Journal of Chinese Buddhist Studies* 30 (2017a): 1-31.
- Radich, Michael. "Problems of Attribution, Style, and Dating Relating to the 'Great Cloud Sūtras' in the Chinese Buddhist Canon (T 387, T 388/S.6916)." In *Buddhist Transformations and Interactions: Papers in Honor of Antonino Forte*, edited by Victor H. Mair, 235-289. Amherst, NY: Cambria Press, 2017b.
- Radich, Michael. "A Triad of Texts from Fifth-Century Southern China: The \**Mahāmāyā-sūtra*, the *Guoqu xianzai yinguo jing*, and a *Mahāparinirvāṇa-sūtra* ascribed to Faxian." *Journal of Chinese Religions*, 46, no. 1 (2018): 1-41.
- Radich, Michael. "Kumārajīva's 'Voice'?" In *China and the World – the World and China: A Transcultural Perspective*, edited by Barbara Mittler, Joachim & Natascha Gentz and Catherine Vance Yeh, 131-145. *Deutsche Ostasienstudien* 37. Gossenberg: Ostasien Verlag, 2019a.
- He Shuqun 何書群 [Michael Radich]. "Zhu Fahu shifou xiuding guo T474? 竺法護是否修訂過 T474?" *Foguang xuebao* 佛光學報, New Series, 5, no. 2 (2019b): 15-38.
- Radich, Michael. "Was the *Mahāparinirvāṇa-sūtra* 大般涅槃經 T7 Translated by 'Faxian'? An Exercise in the Computer-Assisted Assessment of Attributions in the Chinese Buddhist Canon." *Hualin International Journal of Buddhist Studies: E-journal* 2, no. 1 (2019): 229-279. Reprinted in *From Xiangyuan to Ceylon: The Life and Legacy of the Chinese Monk Faxian (227-422)*, edited by Jinhua Chen and Guang Kuan, 374-424. Singapore: World Scholastic Publishers, 2020a.



He Shuqun 何書群 [Michael Radich]. “*Da banniepan jing* (*Mahāparinirvāṇa-sūtra*, T no. 7) shifou you ‘Faxian’ suoyi? Jisuanji xiezhū yiren kanding 《大般涅槃經》 (Mahāparinirvāṇa-sūtra, T no. 7) 是否由「法顯」所譯？計算機協助譯人勘定。” In *Xiantangshan yu Faxian wenhua: Hanseng Faxian (337-422) qi shengping yu yichan guoji yantaohui lunwenji* 僊堂山與法顯文化：漢僧法顯 (337-422) 其生平與遺產國際研討會論文集, edited by Miaojiang 妙江, Chen Jinhua 陳金華, and Ji Yun 紀贇, 337-380. Singapore: World Scholastic Publishers, 2020. Chinese version of Radich 2019/2020.

Radich, Michael and Anālayo Bhikkhu. “Were the *Ekottarika-āgama* 增壹阿含經 T 125 and the *Madhyama-āgama* 中阿含經 T 26 Translated by the Same Person? An Assessment on the Basis of Translation Style.” In *Research on the Madhyama-āgama*, edited by Dhammadinnā, 209-237. Dharma Drum Institute of Liberal Arts Research Series 5. Taipei: Dharma Drum Publishing Corporation, 2017.

### **"Friends of the TACL GUI" wall of fame**

Our warm thanks to all the following people, who helped us in various ways with the development of the TACL GUI: Matthias Arnold, Bai Jinghao 白景皓, Merlin Beutlberger, Marcus Bingenheimer, Jonas Buchholz, Chen Ruixuan, Rafal Felbur, Amanda Jack, Huang Mengji, Jamie Norrish, Jiang Tianren, Lin Qian, Lin Xueni, Qiu Jie, Manuel Sassmann, Sun Hao, Wang Fengyu, Wang Chenyi, Wang Xinru.